Exhibit A

US010452974B1

(12) **United States Patent**
Cosic

(10) **Patent No.: US 10,452,974 B1**
(45) **Date of Patent: Oct. 22, 2019**

(54) **ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION**

(71) Applicant: **Jasmin Cosic**, Miami, FL (US)

(72) Inventor: **Jasmin Cosic**, Miami, FL (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 525 days.

(21) Appl. No.: **15/340,991**

(22) Filed: **Nov. 2, 2016**

(51) **Int. Cl.**
G06N 3/08 (2006.01)
G06N 5/02 (2006.01)

(52) **U.S. Cl.**
CPC ***G06N 3/08*** (2013.01); ***G06N 5/022*** (2013.01)

(58) **Field of Classification Search**
CPC G06N 3/08; G06N 5/022
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,370,707 A 1/1983 Phillips et al.
4,730,315 A 3/1988 Saito et al.
4,860,203 A 8/1989 Corrigan et al.
5,560,011 A 9/1996 Uyama
5,602,982 A 2/1997 Judd et al.
6,026,234 A 2/2000 Hanson et al.
6,088,731 A 7/2000 Kiraly et al.
6,106,299 A 8/2000 Ackermann et al.
6,126,330 A 10/2000 Knight
6,314,558 B1 11/2001 Angel et al.
6,408,262 B1 * 6/2002 Leerberg G06F 11/3608 703/2
6,643,842 B2 11/2003 Angel et al.
6,728,689 B1 4/2004 Drissi et al.
6,735,632 B1 5/2004 Kiraly et al.
6,754,631 B1 6/2004 Din
6,801,912 B2 10/2004 Moskowitz et al.
6,842,877 B2 1/2005 Robarts et al.
6,915,105 B2 7/2005 Masuda
6,973,446 B2 12/2005 Mamitsuka et al.
7,017,153 B2 3/2006 Gouriou et al.
7,052,277 B2 5/2006 Kellman
7,113,946 B2 9/2006 Cosic
7,117,225 B2 10/2006 Cosic
7,222,127 B1 5/2007 Bem et al.
7,240,335 B2 7/2007 Angel et al.
7,249,349 B2 7/2007 Hundt et al.
7,424,705 B2 9/2008 Lewis et al.
7,478,371 B1 1/2009 Gove
7,484,205 B2 1/2009 Venkatapathy
7,565,340 B2 7/2009 Herlocker et al.
7,721,218 B2 5/2010 Awe et al.
(Continued)

OTHER PUBLICATIONS

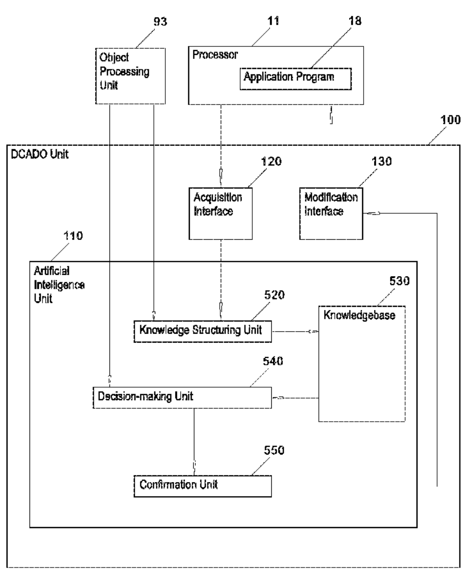
Chen et al. Case-Based Reasoning System and Artificial Neural Networks: A Review Neural Comput & Applic (2001) 10: pp. 264-276, 13 pages.
(Continued)

Primary Examiner — Santiago Garcia

(57) **ABSTRACT**

Aspects of the disclosure generally relate to computing enabled devices and/or systems, and may be generally directed to devices, systems, methods, and/or applications for learning a device's operation in various circumstances, storing this knowledge in a knowledgebase (i.e. neural network, graph, sequences, etc.), and enabling autonomous operation of the device.

20 Claims, 40 Drawing Sheets



(56) **References Cited**

U.S. PATENT DOCUMENTS

7,765,537	B2	7/2010	Havin et al.	
7,797,259	B2	9/2010	Jiang et al.	
7,840,060	B2	11/2010	Podilchuk	
7,925,984	B2	4/2011	Awe et al.	
7,987,144	B1	7/2011	Drissi et al.	
8,005,828	B2	8/2011	Buchner et al.	
8,019,699	B2	9/2011	Baxter	
8,078,556	B2	12/2011	Adi et al.	
8,090,669	B2	1/2012	Shahani et al.	
8,137,112	B2	3/2012	Woolf et al.	
8,166,463	B2	4/2012	Gill et al.	
8,195,674	B1	6/2012	Bem et al.	
8,196,119	B2	6/2012	Gill et al.	
8,244,730	B2	8/2012	Gupta	
8,251,704	B2	8/2012	Woolf et al.	
8,261,199	B2	9/2012	Cradick et al.	
8,266,608	B2	9/2012	Hecht et al.	
8,335,805	B2	12/2012	Cosic	
8,364,612	B2	1/2013	Van Gael et al.	
8,386,401	B2	2/2013	Virkar et al.	
8,397,227	B2	3/2013	Fan et al.	
8,417,740	B2	4/2013	Cosic	
8,464,225	B2	6/2013	Greifeneder	
8,549,359	B2	10/2013	Zheng	
8,572,035	B2	10/2013	Cosic	
8,589,414	B2	11/2013	Waite et al.	
8,595,154	B2 *	11/2013	Breckenridge	G06N 20/00 706/12
8,655,260	B2	2/2014	Yueh et al.	
8,655,900	B2	2/2014	Cosic	
8,667,472	B1	3/2014	Molinari	
8,996,432	B1	3/2015	Fu	
9,047,324	B2	6/2015	Cosic	
9,268,454	B2	2/2016	Hamilton, II et al.	
9,282,309	B1 *	3/2016	Cosic	H04N 9/87
9,298,749	B2	3/2016	Cosic	
9,305,216	B1	4/2016	Mishra	
9,367,806	B1 *	6/2016	Cosic	G06N 5/022
9,443,192	B1 *	9/2016	Cosic	G06N 3/02
9,582,762	B1 *	2/2017	Cosic	G06N 5/04
9,864,933	B1 *	1/2018	Cosic	G06K 9/66
10,102,226	B1 *	10/2018	Cosic	G06F 16/5866
10,102,449	B1 *	10/2018	Cosic	G06K 9/00771
2003/0026588	A1	2/2003	Elder et al.	
2003/0065662	A1	4/2003	Cosic	
2004/0117771	A1	6/2004	Venkatapathy	
2004/0194017	A1	9/2004	Cosic	
2004/0249774	A1	12/2004	Caid et al.	
2004/0267521	A1	12/2004	Cutler et al.	
2005/0149517	A1	7/2005	Cosic	
2005/0149542	A1	7/2005	Cosic	
2005/0240412	A1	10/2005	Fujita	
2005/0245303	A1	11/2005	Graepel et al.	
2005/0289105	A1	12/2005	Cosic	
2006/0047612	A1	3/2006	Stanley et al.	
2006/0184410	A1	8/2006	Ramamurthy et al.	
2006/0190930	A1	8/2006	Hecht et al.	
2006/0265406	A1	11/2006	Chkodrov et al.	
2007/0006159	A1	1/2007	Hecht et al.	
2007/0050606	A1	3/2007	Ferren et al.	
2007/0050719	A1	3/2007	Lui et al.	
2007/0058856	A1	3/2007	Boregowda et al.	
2007/0061735	A1	3/2007	Hoffberg et al.	
2007/0106633	A1	5/2007	Reiner	
2008/0144893	A1	6/2008	Guo et al.	
2008/0215508	A1	9/2008	Hanneman et al.	
2008/0254429	A1	10/2008	Woolf et al.	
2008/0281764	A1	11/2008	Baxter	
2008/0288259	A1	11/2008	Chambers et al.	
2009/0044113	A1	2/2009	Jones et al.	
2009/0067727	A1	3/2009	Hirohata	
2009/0110061	A1	4/2009	Lee et al.	
2009/0131152	A1	5/2009	Busse	
2009/0136095	A1	5/2009	Marcon et al.	
2009/0141969	A1	6/2009	Yu et al.	
2009/0222388	A1	9/2009	Hua et al.	
2009/0287643	A1	11/2009	Corville et al.	
2009/0324010	A1	12/2009	Hou	
2010/0023541	A1	1/2010	Cosic	
2010/0033780	A1	2/2010	Gitter	
2010/0063949	A1	3/2010	Eaton et al.	
2010/0082536	A1	4/2010	Cosic	
2010/0114746	A1	5/2010	Bobbitt et al.	
2010/0138370	A1	6/2010	Wu et al.	
2010/0241595	A1	9/2010	Felsher	
2010/0278420	A1 *	11/2010	Shet	G06K 9/00369 382/156
2011/0007079	A1	1/2011	Perez et al.	
2011/0030031	A1	2/2011	Lussier et al.	
2011/0085734	A1	4/2011	Berg et al.	
2011/0218672	A1	9/2011	Maisonnier et al.	
2011/0270794	A1	11/2011	Drory et al.	
2012/0150773	A1	6/2012	DiCorpo et al.	
2012/0167057	A1 *	6/2012	Schmich	G06F 11/3644 717/130
2012/0284026	A1	11/2012	Cardillo et al.	
2012/0290347	A1 *	11/2012	Elazouni	G06Q 10/00 705/7.12
2013/0007532	A1	1/2013	Miller et al.	
2013/0156345	A1	6/2013	Shmunk	
2013/0159021	A1	6/2013	Felsher	
2013/0218932	A1	8/2013	Cosic	
2013/0226974	A1	8/2013	Cosic	
2013/0238533	A1	9/2013	Virkar et al.	
2013/0278501	A1	10/2013	Bulzacki	
2013/0278631	A1	10/2013	Border et al.	
2014/0052717	A1	2/2014	Franks et al.	
2014/0075249	A1	3/2014	Sato et al.	
2014/0161250	A1	6/2014	Leeds et al.	
2014/0177946	A1 *	6/2014	Lim	G06K 9/4614 382/156
2014/0207580	A1	7/2014	Minnis et al.	
2014/0211988	A1	7/2014	Fan et al.	
2015/0006171	A1	1/2015	Westby et al.	
2015/0039304	A1	2/2015	Wein	
2015/0055821	A1	2/2015	Fotland	
2015/0264306	A1	9/2015	Marilly et al.	
2015/0269415	A1	9/2015	Gelbman	
2015/0310041	A1	10/2015	Kier et al.	
2015/0324685	A1	11/2015	Bohn et al.	
2015/0339213	A1 *	11/2015	Lee	G06F 11/3664 717/125
2016/0077166	A1 *	3/2016	Morozov	G01C 19/00 702/150
2016/0140999	A1	5/2016	Cosic	
2016/0142650	A1	5/2016	Cosic	
2016/0167226	A1 *	6/2016	Schnittman	B25J 5/00 382/153
2016/0243701	A1 *	8/2016	Gildert	G05B 19/42
2016/0246819	A1	8/2016	Cosic	
2016/0246850	A1	8/2016	Cosic	
2016/0246868	A1	8/2016	Cosic	
2016/0274187	A1 *	9/2016	Menon	G01R 31/31718
2016/0292185	A1	10/2016	Cosic	
2016/0328480	A1	11/2016	Owens et al.	
2017/0161958	A1 *	6/2017	Eilat	G02B 27/017
2018/0101146	A1 *	4/2018	Hariharan	G05B 13/0265
2018/0120813	A1 *	5/2018	Coffman	G05B 19/4097

OTHER PUBLICATIONS

John J. Grefenstette, Connie Loggia Ramsey, Alan C. Schultz, Learning Sequential Decision Rules Using Simulation Models and Competition, 1990, Navy Center for Applied Research in Artificial Intelligence, Naval Research Laboratory, Washington, DC, 27 pages.

Alan C. Schultz, John J. Grefenstette, Using a Genetic Algorithm to Learn Behaviors for Autonomous Vehicles, 1992, Navy Center for Applied Research in Artificial Intelligence, Naval Research Laboratory, Washington, DC, 12 pages.

Koppula et al., "Anticipating human activities using object affordances for reactive robotic response", IEEE TRAMI 2016, published May 5, 2015, 16 pages.

(56) **References Cited**

OTHER PUBLICATIONS

Orme, "System design tips for entry level smartphones—part 3", found online at "https://community.arm.com/processors/b/blog/posts/system-design-tips-for-entry-level-smartphones-part-3", Oct. 21, 2013, 12 pages.
Nowostawski, Mariusz et al., "Dynamic Demes Parallel Genetic Algorithm", May 13, 1999, 6 pages.
Stack Overflow, How can I quantify difference between two images?, accessed Aug. 2, 2017 at https://stackoverflow.com/questions/189943/how-can-i-quantify-difference-belween-two-images, 8 pages.
Skolicki, Zbigniew et al., "The Influence of Migration Sizes and Intervals on Island Models", Jun. 29, 2005, 8 pages.
Vahid Lari, et al., "Decentralized dynamic resource management support for massively parallel processor arrays", Sep. 11, 2011, 8 pages.
Vahid Lari, et al., "Distributed resource reservation in massively parallel processor arrays", May 16, 2011, 4 pages.
Mohsen Hayati and Yazdan Shirvany, "Artificial Neural Network Approach for Short Term Load Forecasting for Illam Region", Jan. 2007, 5 pages.
Hasim Sak, Andrew Senior, and Francoise Beaufays, "Long Short-Term Memory Recurrent Neural Network Architectures for Large Scale Acoustic Modeling", Jan. 2014, 5 pages.
Jorg Walter and Klaus Schulten, "Implementation of self-organizing neural networks for visuo-motor control of an industrial robot", Jan. 1993, 10 pages.
Heikki Hyotyniemi and Aamo Lehtola, "A Universal Relation Database Interface for Knowledge Based Systems", Apr. 1991, 5 pages.
Mrissa, Michael, et al. "An avatar architecture for the web of things." IEEE Internet Computing 19.2 (2015): 30-38., 9 pages.
Luck, Michael, and Ruth Aylett. "Applying artificial intelligence to virtual reality: Intelligent virtual environments." Applied Artificial Intelligence 14.1 (2000): 3-32., 30 pages.
Terdjimi, Mehdi, et al. "An avatar-based adaptation workflow for the web of things." Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), 2016 IEEE 25th International Conference on. IEEE, 2016., 6 pages.
Bogdanovych, Anton, et al. "Authentic interactive reenactment of cultural heritage with 3D virtual worlds and artificial intelligence." Applied Artificial Intelligence 24.6 (2010): 617-647., 32 pages.
Hernandez, Marco E. Perez, and Stephan Reiff-Marganiec. "Autonomous and self controlling smart objects for the future internet." Future Internet of things and cloud (FiCloud), 2015 3rd international conference on. IEEE, 2015., 8 pages.
Medini et al., "Building a Web of Things with Avatars", Managing the Web of Things (2017), 30 pages.
About | OpenCV, retrieved from <URL: http://opencv.org/about.html> on Dec. 13, 2014, 1 pages.
Animetrics, Inc. | 3D Facial Recognition, retrieved from <URL: http://animetrics.com/> on Dec. 13, 2014, 2 pages.
Bag-of-words model, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 2 pages.
Beier—Neely morphing algorithm, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 1 pages.
Computer vision, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 7 pages.
Convolutional neural network, retrieved from <URL: http://wikipedia.com> on Nov. 11, 2015, 5 pages.
Decimation (signal processing), retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 3 pages.
Digital image processing, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 3 pages.
Dissolve (filmmaking), retrieved from <URL: http://wikipedia.com> on Nov. 11, 2015, 2 pages.
Facial recognition system, retrieved from <URL: http://wikipedia.com> on Nov. 11, 2015, 6 pages.
Feature detection (computer vision), retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 3 pages.
Feature extraction, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 2 pages.
Gesture recognition, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 5 pages.
GrabCut, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 1 pages.
Image processing, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 2 pages.
Image segmentation, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 12 pages.
Image warping, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 2 pages.
Inbetweening, retrieved from <URL: http://wikipedia.com> on Nov. 11, 2015, 2 pages.
Interpolation, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 5 pages.
Language model, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 4 pages.
List of speech recognition software, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 4 pages.
Livewire Segmentation Technique, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 2 pages.
Morphing, retrieved from <URL: http://wikipedia.com> on Nov. 11, 2015, 2 pages.
Motion estimation, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 2 pages.
Muse: Face Morph || Mesh Warping, retrieved from <URL: http://alexwolfe.blogspot.com/2011/10/face-morph-mesh-warping.html> on Nov. 19, 2015, 2 pages.
N-gram, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 5 pages.
Optical flow, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 4 pages.
Outline of object recognition, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 7 pages.
Phonotactics, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 3 pages.
Recurrent neural network, retrieved from <URL: http://wikipedia.com> on Nov. 11, 2015, 5 pages.
Sample rate conversion, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 2 pages.
Simple interactive object extraction, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 2 pages.
Speech recognition, retrieved from <URL: http://wikipedia.com> on Nov. 11, 2015, 12 pages.
Speech segmentation, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 4 pages.
Welcome to recognize-speech.com, retrieved from <URL: http://recognize-speech.com/> on Oct. 18, 2015, 1 pages.
Introduction Speech, retrieved from <URL: http://recognize-speech.com/speech> on Oct. 18, 2015, 1 pages.
Preprocessing, retrieved from <URL: http://recognize-speech.com/preprocessing> on Oct. 18, 2015, 4 pages.
Feature Extraction, retrieved from <URL: http://recognize-speech.com/feature-extraction> on Oct. 18, 2015, 3 pages.
Acoustic model, retrieved from <URL: http://recognize-speech.com/acoustic-model> on Oct. 18, 2015, 2 pages.
Video content analysis, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 2 pages.
Video tracking, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 2 pages.
Andrej Karpathy, Thomas Leung, George Toderici, Rahul Sukthankar, Sanketh Shetty, Li Fei-Fei, Large-scale Video Classification with Convolutional Neural Networks, Apr. 14, 2014, 8 pages, Stanford University.
Karen Simonyan, Andrew Zisserman, Two-Stream Convolutional Networks for Action Recognition in Videos, Nov. 13, 2014, 11 pages, University of Oxford.
About Event Tracing, retrieved from <URL: http://msdn.microsoft.com/en-us/library/aa363668(d=default,l=en-us, v=vs.85).aspx> on Jan. 12, 2014, 2 pages.

(56) **References Cited**

OTHER PUBLICATIONS

Add dynamic Java code to your application, retrieved from <URL: http://www.javaworld.com/article/2071777/design-patterns/add-dynamic-java-code-to-your-application.html> on May 13, 2014, 6 pages.
BCEL, retrieved from <URL: http://commons.apache.org> on May 13, 2014, 2 pages.
Artificial intelligence, retrieved from <URL: http://wikipedia.com> on May 21, 2014, 28 pages.
Artificial neural network, retrieved from <URL: http://wikipedia.com> on May 21, 2014, 12 pages.
Branch (computer science), retrieved from <URL: http://wikipedia.com> on May 6, 2014, 2 pages.
Branch trace, retrieved from <URL: http://wikipedia.com> on May 28, 2014, 2 pages.
C Function Call Conventions and the Stack, retrieved from <URL: http://www.csee.umbc.edu/~chang/cs313.s02/stack.shtml> on Jan. 11, 2014, 4 pages.
Call stack, retrieved from <URL: http://wikipedia.com> on Jan. 11, 2014, 5 pages.
Program compilation and execution flow, retrieved from <URL: http://cs.stackexchange.com/questions/6187/program-compilation-and-execution-flow> on Jan. 9, 2014, 2 pages.
Control unit, retrieved from <URL: http://wikipedia.com> on May 25, 2014, 2 pages.
Create dynamic applications with javax.tools, retrieved from <URL: http://www.ibm.com/developerworks/library/j-jcomp/> on May 13, 2014, 11 pages.
Creating a Debugging and Profiling Agent with JVMTI, retrieved from <URL: http://www.oracle.com/technetwork/articles/java/jvmti-136367.html> on Jan. 11, 2014, 18 pages.
Decision tree learning, retrieved from <URL: http://wikipedia.com> on Jun. 24, 2014, 5 pages.
Dynamic Bytecode Instrumentation, retrieved from <URL: http://www.drdobbs.com/tools/dynamic-bytecode-instrumentation/184406433> on Apr. 26, 2014, 6 pages.
Dynamic programming language, retrieved from <URL: http://wikipedia.com> on May 7, 2014, 4 pages.
Dynamic recompilation, retrieved from <URL: http://wikipedia.com> on May 19, 2014, 4 pages.
Eval(), retrieved from <URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/eval> on May 7, 2014, 13 pages.
Function, retrieved from <URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Function> on May 7, 2014, 11 pages.
FunctionEnter2 Function, retrieved from <URL: https://msdn.microsoft.com/en-us/library/aa964981(d=default,I=en-us, v=vs.110).aspx> on Mar. 19, 2014, 2 pages.
FunctionLeave2 Function, retrieved from <URL: https://msdn.microsoft.com/en-us/library/aa964942(d=default,I=en-us, v=vs.110).aspx> on Mar. 19, 2014, 1 pages.
Heuristic, retrieved from <URL: http://wikipedia.com> on May 22, 2014, 5 pages.
How to: Configure Trace Switches, retrieved from <URL: https://msdn.microsoft.com/en-us/library/vstudio/t06xyy08(d=default,I=en-us,v=vs.100).aspx> on Mar. 19, 2014, 3 pages.
Indirect branch, retrieved from <URL: http://wikipedia.com> on May 5, 2014, 1 pages.
Instruction cycle, retrieved from <URL: http://wikipedia.com> on Apr. 30, 2014, 2 pages.
Instruction set, retrieved from <URL: http://wikipedia.com> on Apr. 29, 2014, 6 pages.
Interrupt, retrieved from <URL: http://wikipedia.com> on May 15, 2014, 5 pages.
Introduction to Instrumentation and Tracing, retrieved from <URL: http://msdn.microsoft.com/en-us/library/aa983649(d=default,I=en-us,v=vs.71).aspx> on Jan. 10, 2014, 2 pages.
Java bytecode, retrieved from <URL: http://wikipedia.com> on May 27, 2014, 4 pages.
Package java.util.logging, retrieved from <URL: http://docs.oracle.com/javase/1.5.0/docs/api/java/util/logging/package-summary.html> on Mar. 19, 2014, 2 pages.
Javac—Java programming language compiler, retrieved from <URL: http://docs.oracle.com/javase/7/docs/technotes/tools/windows/javac.html> on May 13, 2014, 12 pages.
Interface JavaCompiler, retrieved from <URL: http://wikipedia.com> on May 13, 2014, 4 pages.
Javassist, retrieved from <URL: http://wikipedia.com> on May 13, 2014, 2 pages.
Just-in-time compilation, retrieved from <URL: http://wikipedia.com> on May 19, 2014, 5 pages.
JVM Tool Interface, retrieved from <URL: http://docs.oracle.com/javase/7/docs/plafform/jvmti/jvmti.html> on Apr. 26, 2014, 127 pages.
Machine code, retrieved from <URL: http://wikipedia.com> on Apr. 30, 2014, 3 pages.
Metaprogramming, retrieved from <URL: http://wikipedia.com> on May 8, 2014, 3 pages.
Microsoft.VisualBasic.Logging Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/ms128122(d=default,I=en-us,v=vs.110).aspx> on Jan. 13, 2014, 1 pages.
Pin (computer program), retrieved from <URL: http://wikipedia.com> on May 6, 2014, 3 pages.
Processor register, retrieved from <URL: http://wikipedia.com> on Apr. 29, 2014, 4 pages.
Profiling (computer programming), retrieved from <URL: http://wikipedia.com> on Jan. 10, 2014, 4 pages.
Profiling Overview, retrieved from <URL: http://msdn.microsoft.com/en-us/library/bb384493(d=default,I=en-us, v=vs.110).aspx> on Jan. 12, 2014, 6 pages.
Program counter, retrieved from <URL: http://wikipedia.com> on Apr. 24, 2014, 2 pages.
Ptrace, retrieved from <URL: http://wikipedia.com> on Apr. 24, 2014, 2 pages.
Reflection (computer programming), retrieved from <URL: http://wikipedia.com> on May 7, 2014, 5 pages.
Class Runtime, retrieved from <URL: http://docs.oracle.com/javase/7/docs/api/java/lang/Runtime.html> on Jan. 10, 2014, 10 pages.
Self-modifying code, retrieved from <URL: http://wikipedia.com> on Jan. 12, 2014, 6 pages.
Subjective logic, retrieved from <URL: http://wikipedia.com> on May 22, 2014, 5 pages.
System.CodeDom.Compiler Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/z6b99ydt(d=default,I=en-us,v=vs.110).aspx> on May 13, 2014, 2 pages.
System.Reflection.Emit Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/xd5fw18y(d=default =en-us,v=vs.110).aspx> on May 13, 2014, 2 pages.
Tracing (software), retrieved from <URL: http://wikipedia.com> on Jan. 10, 2014, 3 pages.
Tree (data structure), retrieved from <URL: http://wikipedia.com> on Jun. 24, 2014, 6 pages.
PTRACE(2), retrieved from <URL: http://unixhelp.ed.ac.uk/CGI/man-cgi?ptrace> on Mar. 19, 2014, 5 pages.
Wevtutil, retrieved from <URL: http://technet.microsoft.com/en-us/library/cc732848(d=default,I=en-us,v=ws.11).aspx> on Apr. 28, 2014, 5 pages.
Intel Processor Trace, retrieved from <URL: https://software.intel.com/en-us/blogs/2013/09/18/processor-tracing> on Apr. 28, 2014, 3 pages.
Younghoon Jung, Java Dynamics Reflection and a lot more, Oct. 10, 2012, 55 pages, Columbia University.
Amitabh Srivastava, Alan Eustace, Atom A System for Building Customized Program Analysis Tools, May 3, 2004, 12 pages.
Mathew Smithson, Kapil Anand, Aparna Kotha, Khaled Elwazeer, Nathan Giles, Rajeev Barua, Binary Rewriting without Relocation Information, Nov. 10, 2010, 11 pages, University of Maryland.
Marek Olszewski, Keir Mierte, Adam Czajkowski, Angela Demle Brown, JIT Instrumentation—A Novel Approach to Dynamically

(56) **References Cited**

OTHER PUBLICATIONS

Instrument Operating Systems, Feb. 12, 2007, 14 pages, University of Toronto.

* cited by examiner

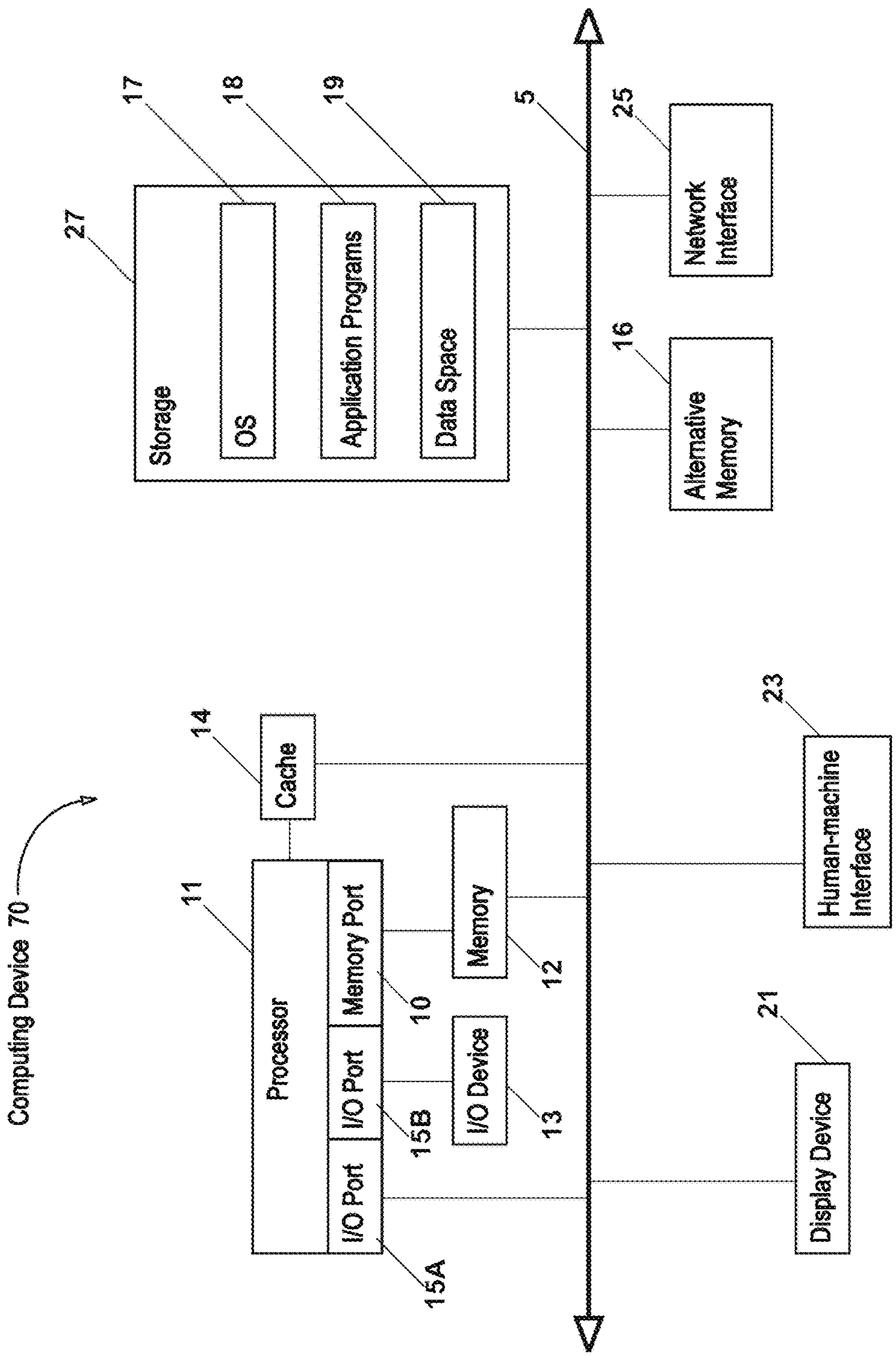

Computing Device 70
Storage
OS
Application Programs
Data Space
Network Interface
Alternative Memory
Cache
Processor
Memory Port
I/O Port
I/O Port
Memory
I/O Device
Human-machine Interface
Display Device
27
17
18
19
5
25
16
14
11
10
12
13
15B
15A
23
21

FIG. 1

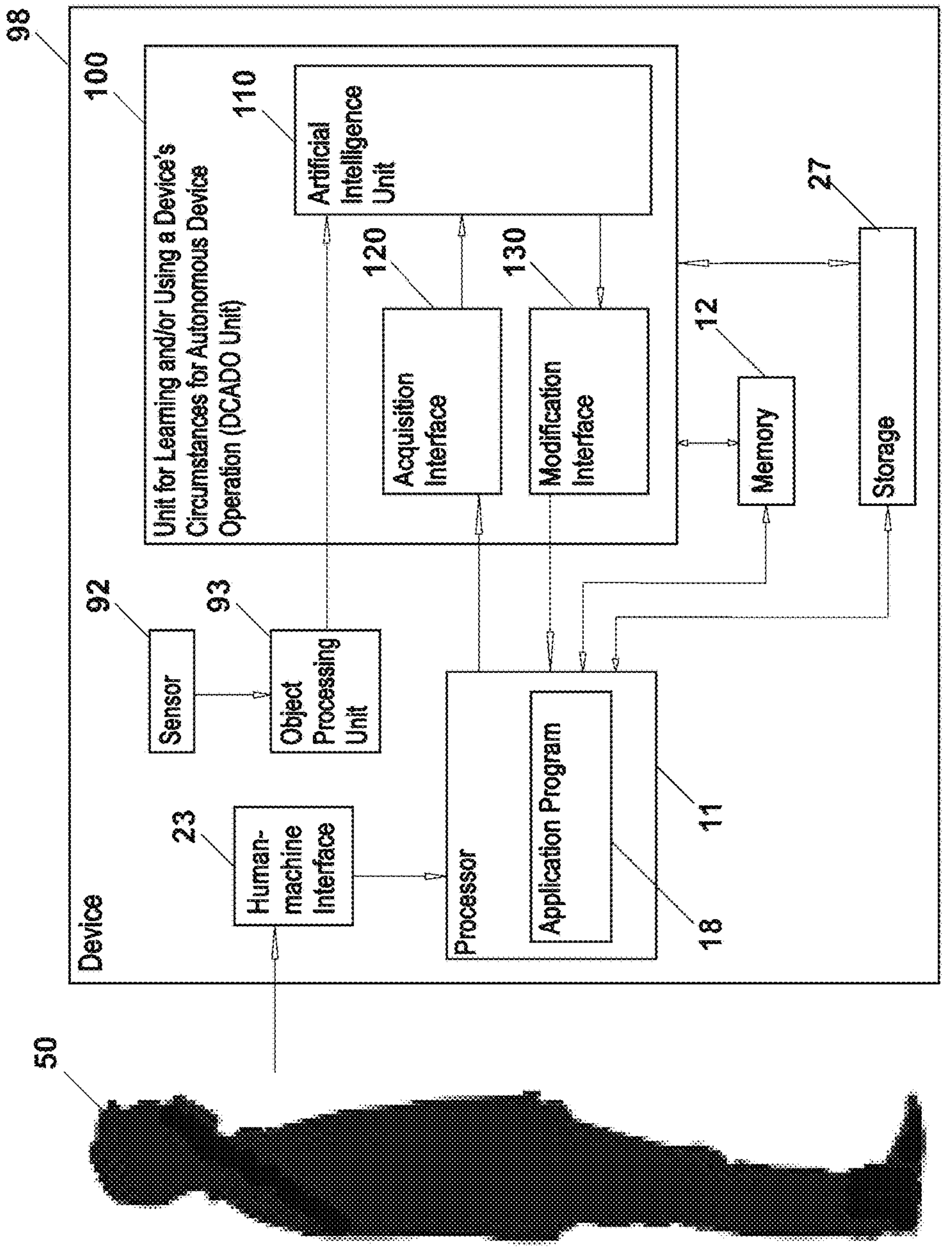

98
Device
100
Unit for Learning and/or Using a Device's Circumstances for Autonomous Device Operation (DCADO Unit)
110
Artificial Intelligence Unit
120
Acquisition Interface
130
Modification Interface
12
Memory
27
Storage
92
Sensor
93
Object Processing Unit
23
Human-machine Interface
Processor
Application Program
11
18
50

FIG. 2

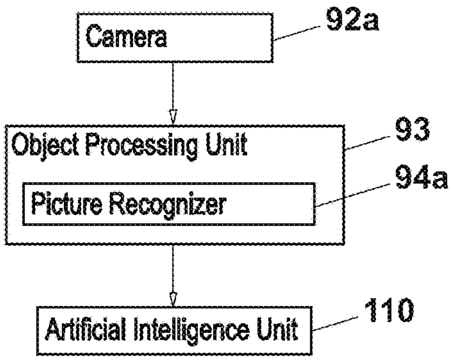


FIG. 3A

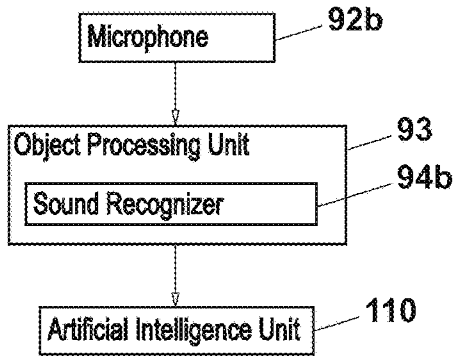


FIG. 3B

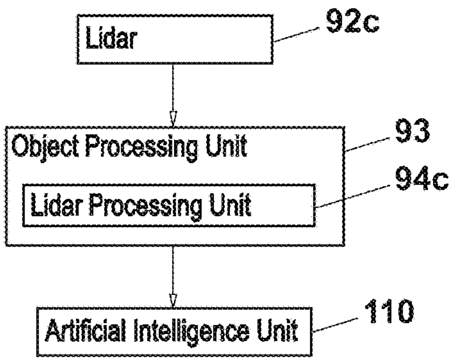


FIG. 3C

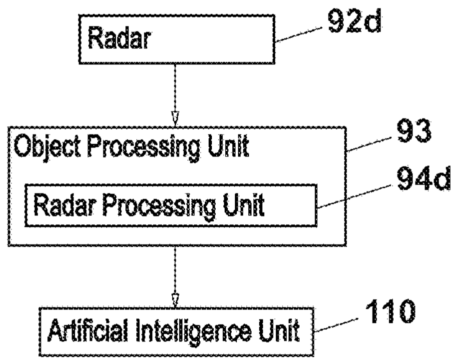


FIG. 3D

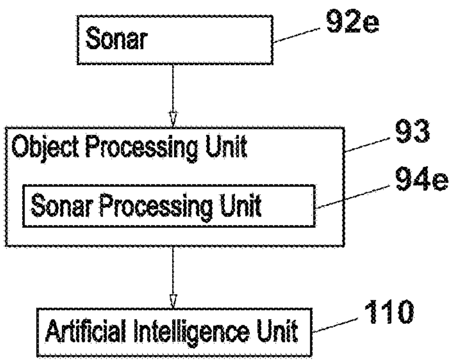
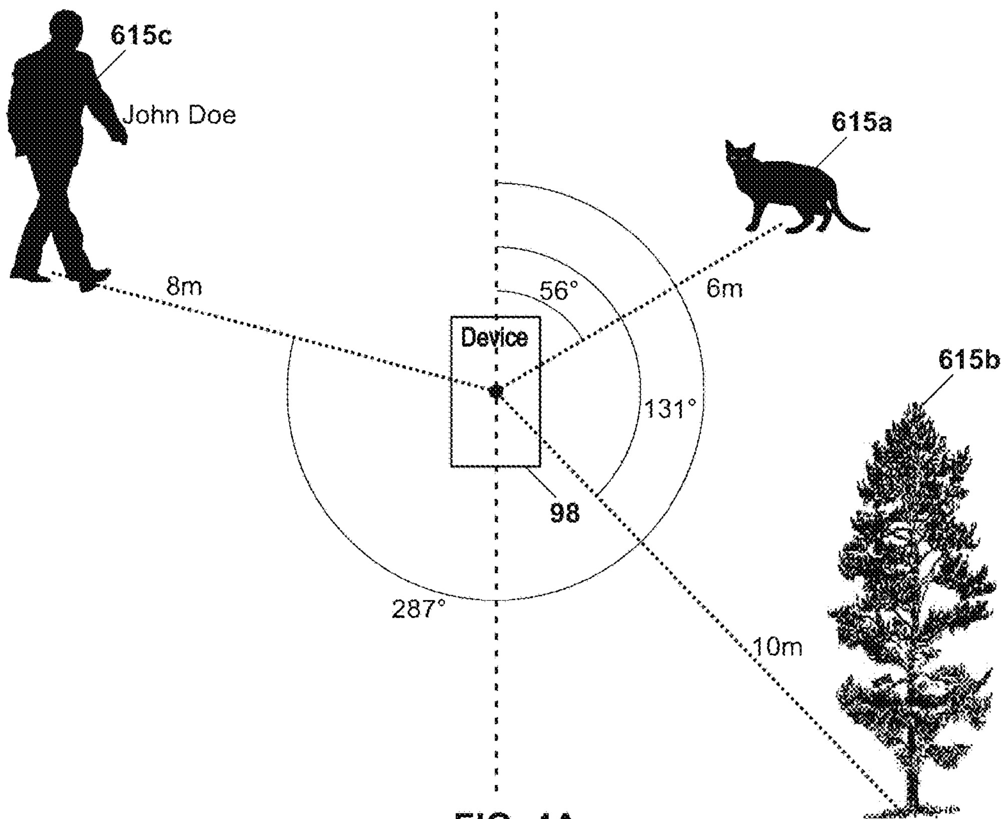
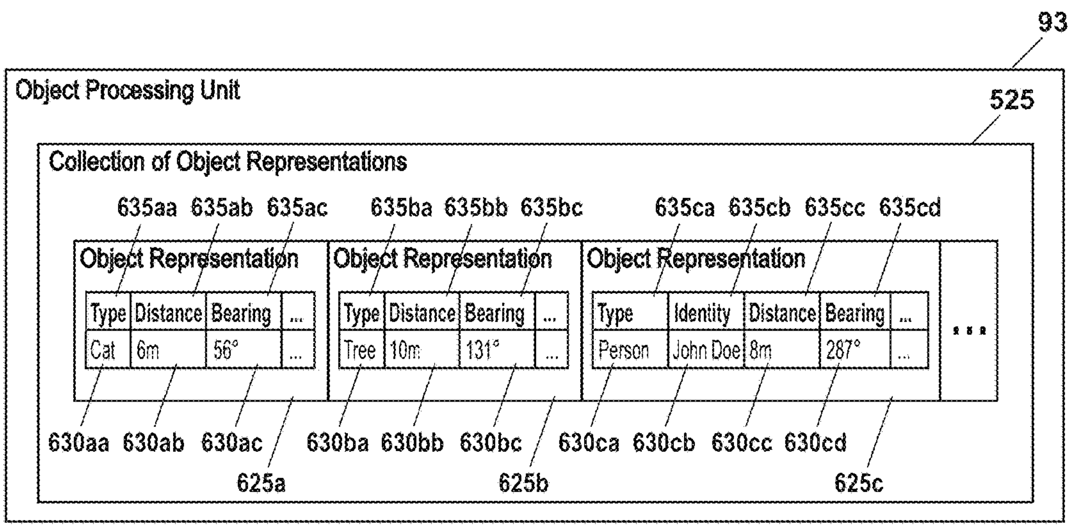


FIG. 3E



615c
John Doe
8m
615a
56°
6m
Device
131°
615b
98
287°
10m

FIG. 4A



93
Object Processing Unit
525
Collection of Object Representations
635aa 635ab 635ac
635ba 635bb 635bc
635ca 635cb 635cc 635cd
Object Representation
Type | Distance | Bearing | ...
Cat | 6m | 56° | ...
Object Representation
Type | Distance | Bearing | ...
Tree | 10m | 131° | ...
Object Representation
Type | Identity | Distance | Bearing | ...
Person | John Doe | 8m | 287° | ...
...
630aa 630ab 630ac
630ba 630bb 630bc
630ca 630cb 630cc 630cd
625a
625b
625c

FIG. 4B

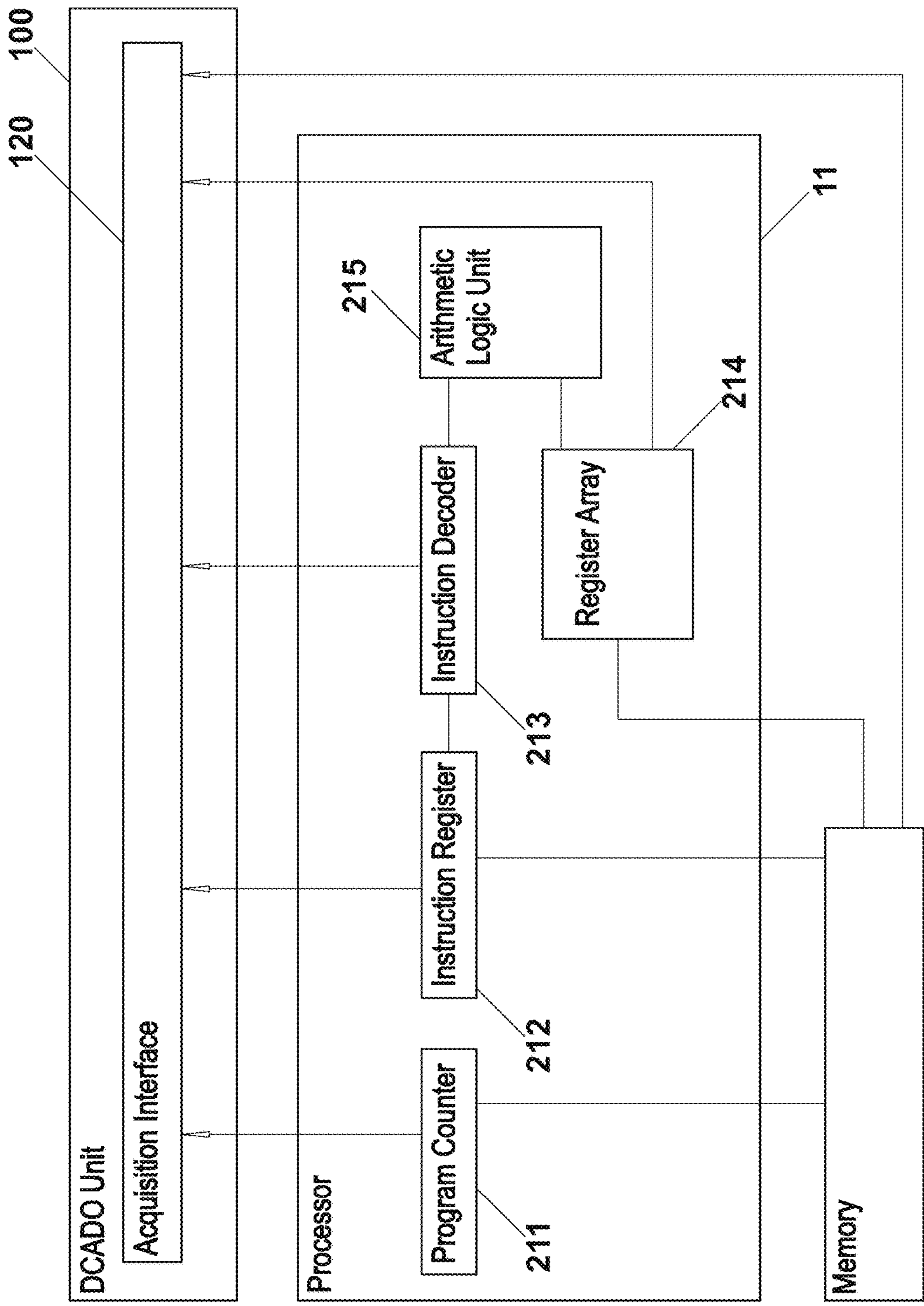

100
120
DCADO Unit
Acquisition Interface
Processor
Program Counter
211
Instruction Register
212
Instruction Decoder
213
Arithmetic Logic Unit
215
Register Array
214
11
Memory
12

FIG. 5

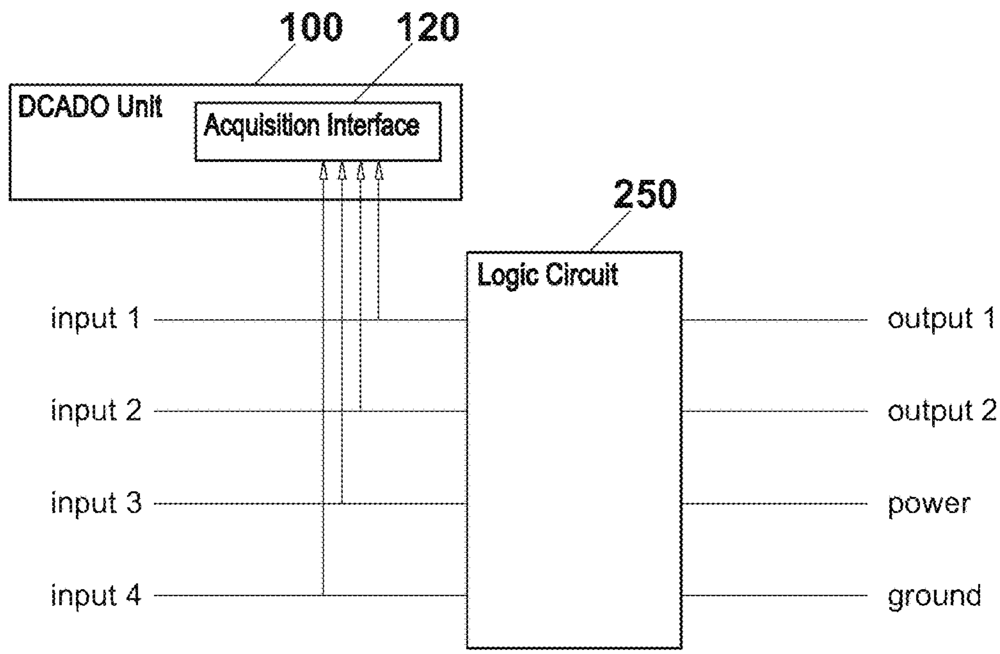

100
120
DCADO Unit
Acquisition Interface
250
Logic Circuit
input 1
input 2
input 3
input 4
output 1
output 2
power
ground

FIG. 6A

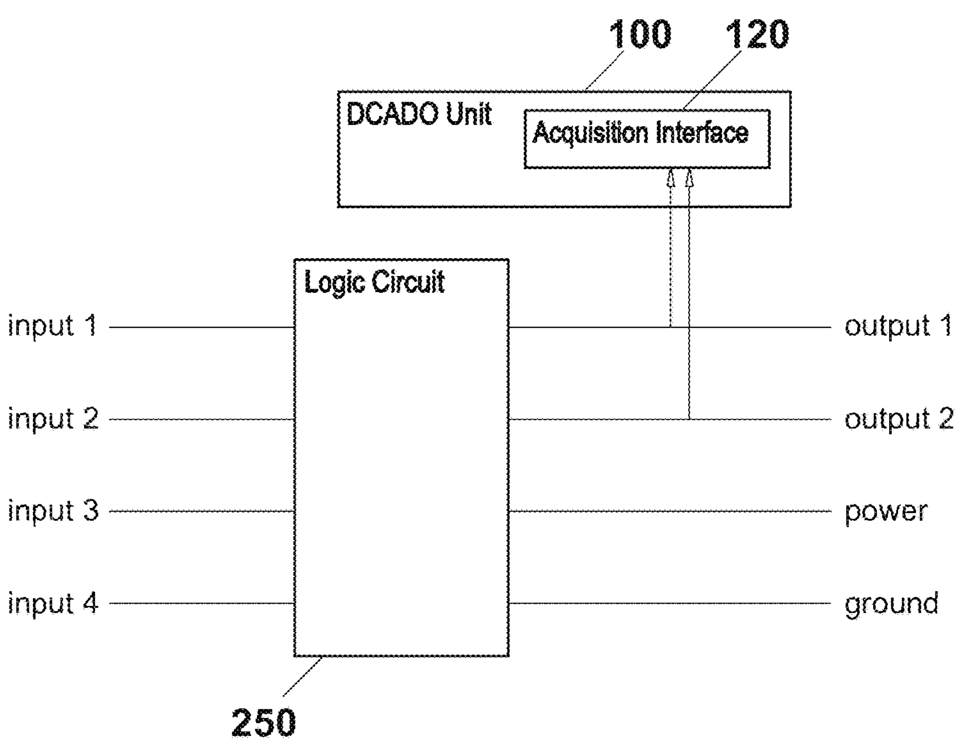

100
120
DCADO Unit
Acquisition Interface
Logic Circuit
input 1
input 2
input 3
input 4
output 1
output 2
power
ground
250

FIG. 6B

526

Instruction Set

Function (Parameter1, Parameter2, Parameter3, ...)

FIG. 7A

526

Instruction Set

SELECT Col1, Col2 ... FROM Tbl1, Tbl2 ... ;

FIG. 7B

526

Instruction Set

iconst_Value

FIG. 7C

526

Instruction Set

add Register1 Register2

FIG. 7D

526

Instruction Set

000000 00001 00010 00110 00000 100000

FIG. 7E

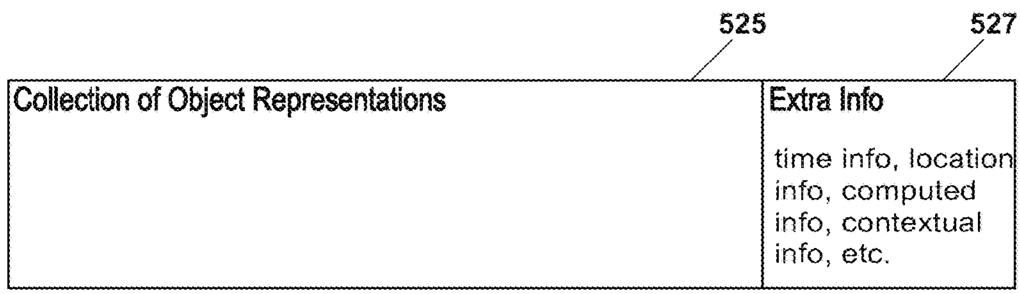

525
527
Collection of Object Representations
Extra Info
time info, location info, computed info, contextual info, etc.

FIG. 8A

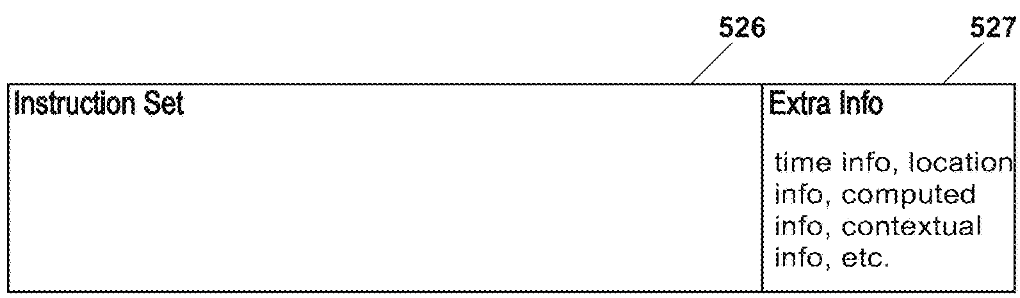

526
527
Instruction Set
Extra Info
time info, location info, computed info, contextual info, etc.

FIG. 8B

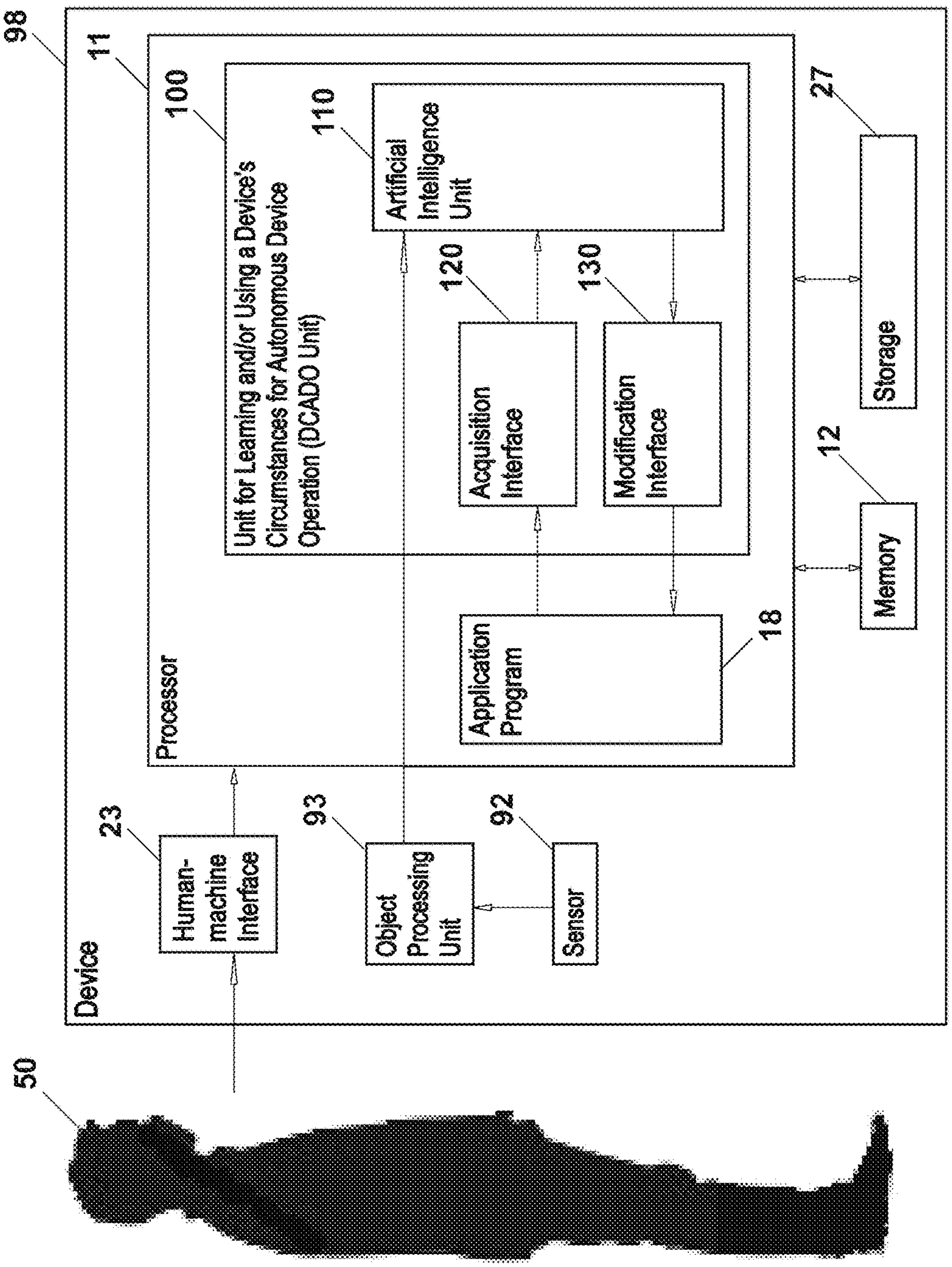

98
Device
11
Processor
100
Unit for Learning and/or Using a Device's Circumstances for Autonomous Device Operation (DCADO Unit)
110
Artificial Intelligence Unit
120
Acquisition Interface
130
Modification Interface
18
Application Program
27
Storage
12
Memory
23
Human-machine Interface
93
Object Processing Unit
92
Sensor
50

FIG. 9

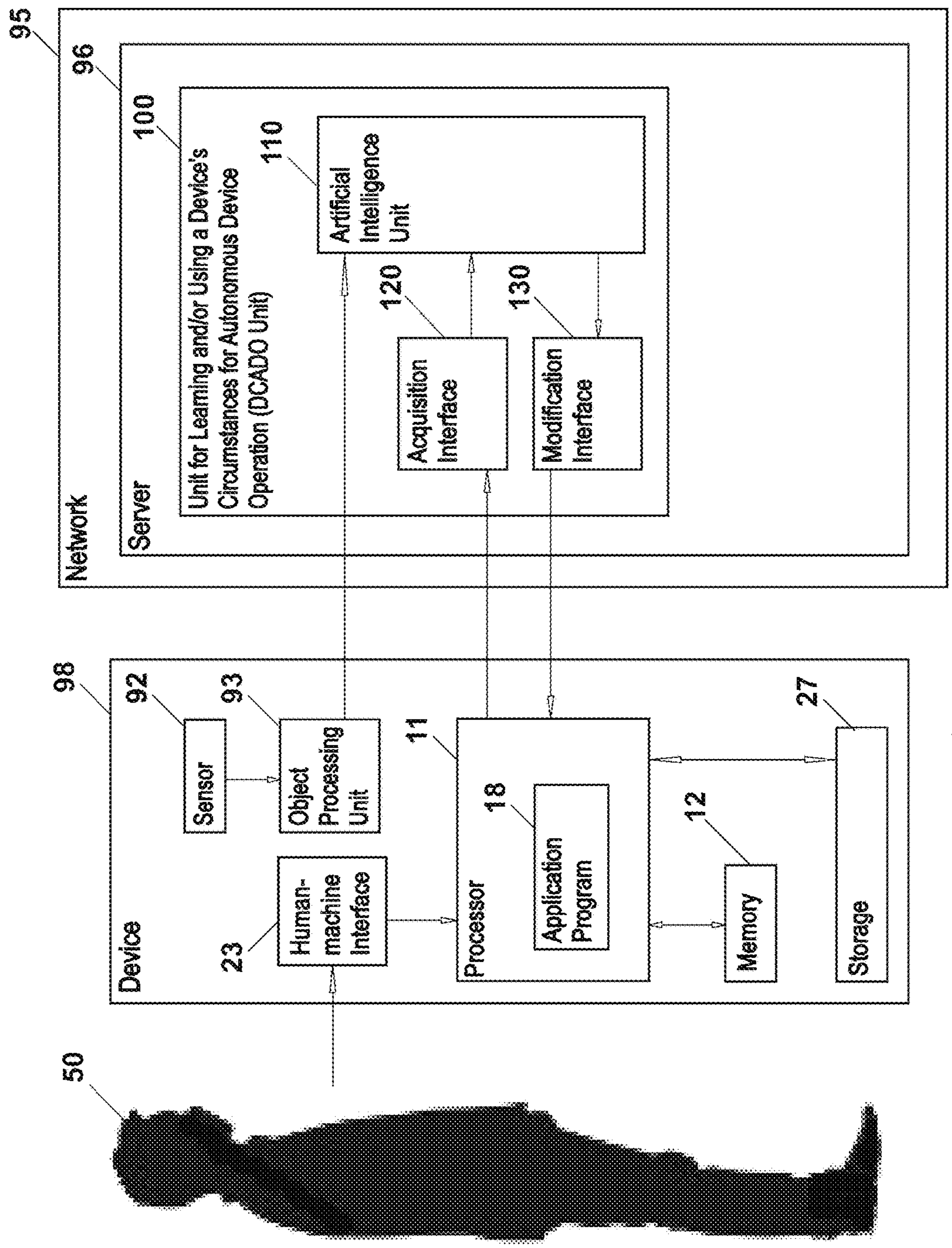

95
Network
96
Server
100
Unit for Learning and/or Using a Device's Circumstances for Autonomous Device Operation (DCADO Unit)
110
Artificial Intelligence Unit
120
Acquisition Interface
130
Modification Interface
98
Device
92
Sensor
93
Object Processing Unit
23
Human-machine Interface
11
Processor
18
Application Program
12
Memory
27
Storage
50

FIG. 10

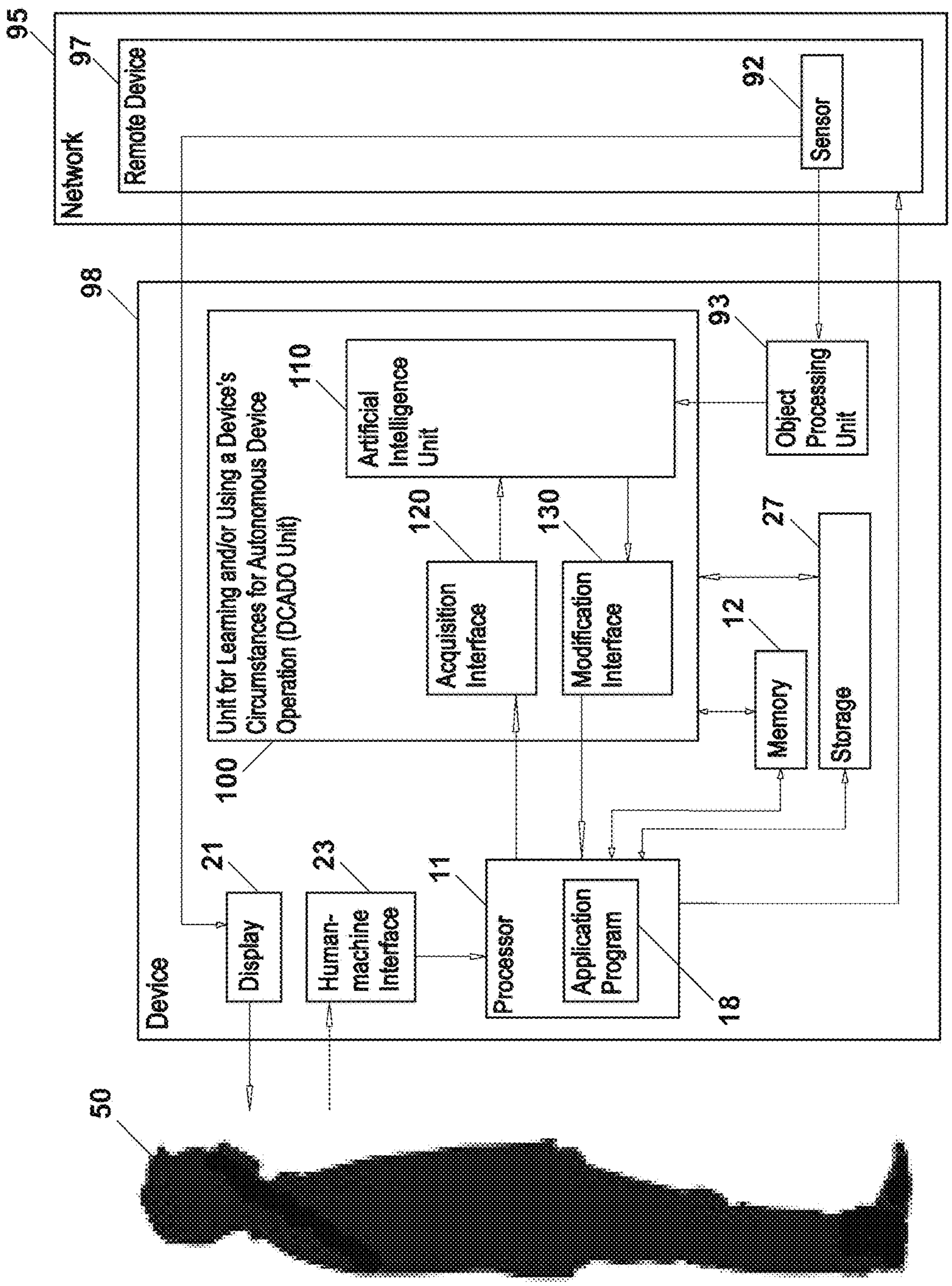

95
97
Network
Remote Device
92
Sensor
98
Device
100
Unit for Learning and/or Using a Device's Circumstances for Autonomous Device Operation (DCADO Unit)
110
Artificial Intelligence Unit
93
Object Processing Unit
120
Acquisition Interface
130
Modification Interface
27
Storage
12
Memory
21
Display
23
Human-machine Interface
11
Processor
Application Program
18
50

FIG. 11

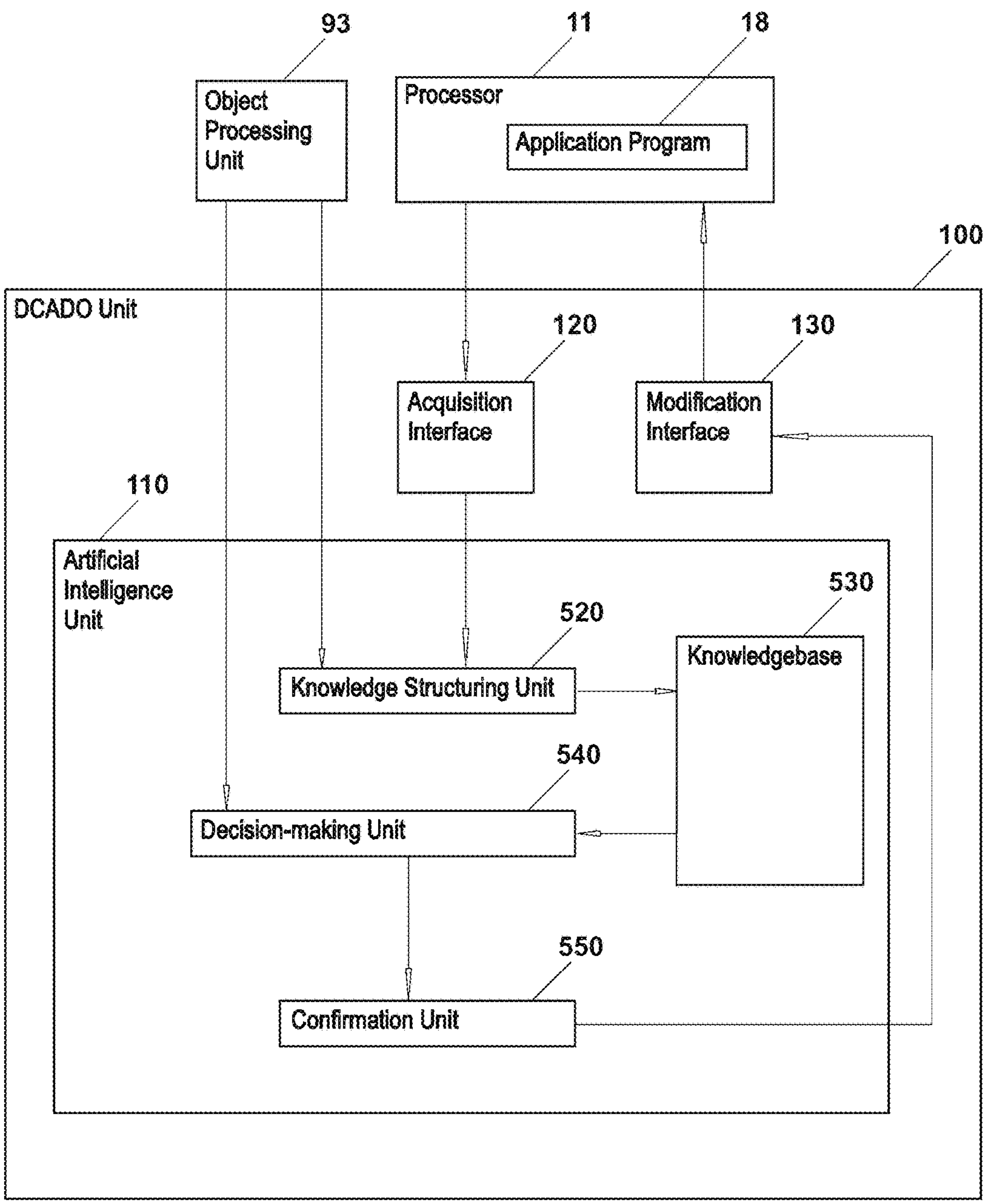
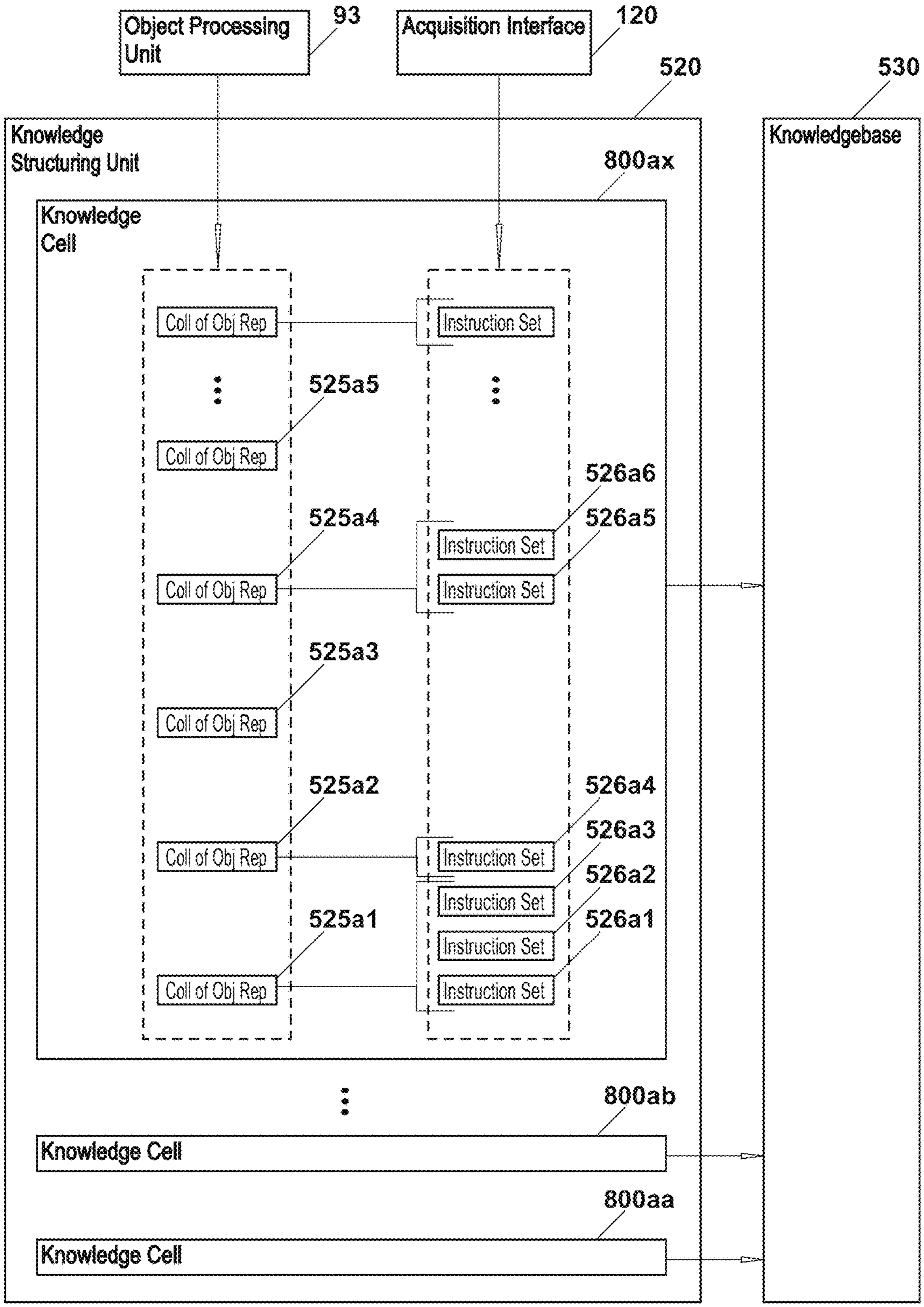
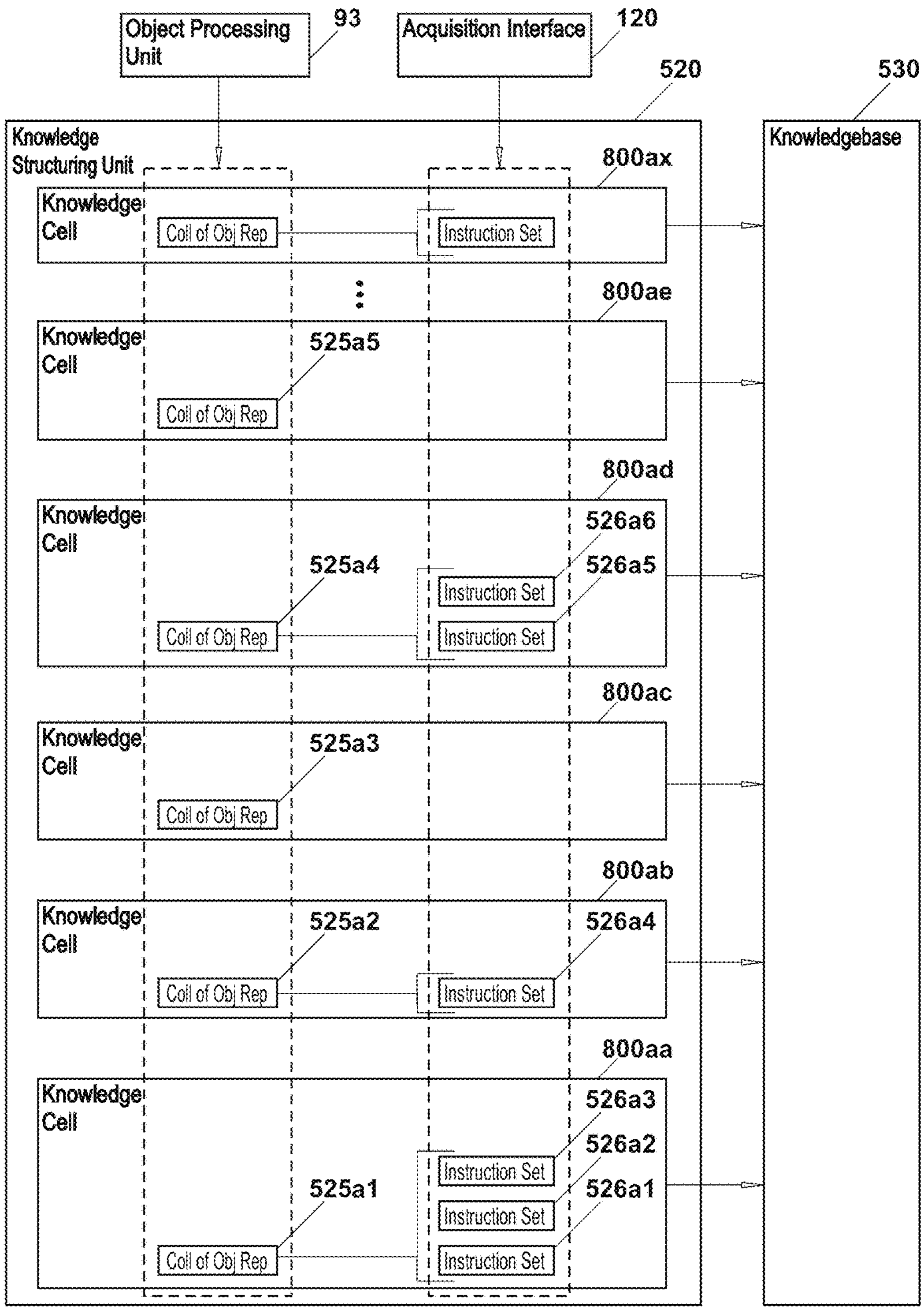

93
11
18
Object Processing Unit
Processor
Application Program
100
DCADO Unit
120
130
Acquisition Interface
Modification Interface
110
Artificial Intelligence Unit
530
520
Knowledgebase
Knowledge Structuring Unit
540
Decision-making Unit
550
Confirmation Unit

FIG. 12



Object Processing Unit
93
Acquisition Interface
120
520
530
Knowledge Structuring Unit
Knowledgebase
800ax
Knowledge Cell
Coll of Obj Rep
Instruction Set
525a5
Coll of Obj Rep
526a6
525a4
526a5
Instruction Set
Coll of Obj Rep
Instruction Set
525a3
Coll of Obj Rep
525a2
526a4
526a3
Coll of Obj Rep
Instruction Set
526a2
Instruction Set
525a1
526a1
Instruction Set
Coll of Obj Rep
Instruction Set
800ab
Knowledge Cell
800aa
Knowledge Cell

FIG. 13



Object Processing Unit
93
Acquisition Interface
120
520
530
Knowledge Structuring Unit
Knowledgebase
800ax
Knowledge Cell
Coll of Obj Rep
Instruction Set
800ae
Knowledge Cell
525a5
Coll of Obj Rep
800ad
Knowledge Cell
526a6
525a4
526a5
Instruction Set
Coll of Obj Rep
Instruction Set
800ac
Knowledge Cell
525a3
Coll of Obj Rep
800ab
Knowledge Cell
525a2
526a4
Coll of Obj Rep
Instruction Set
800aa
Knowledge Cell
526a3
526a2
525a1
526a1
Instruction Set
Instruction Set
Coll of Obj Rep
Instruction Set

FIG. 14

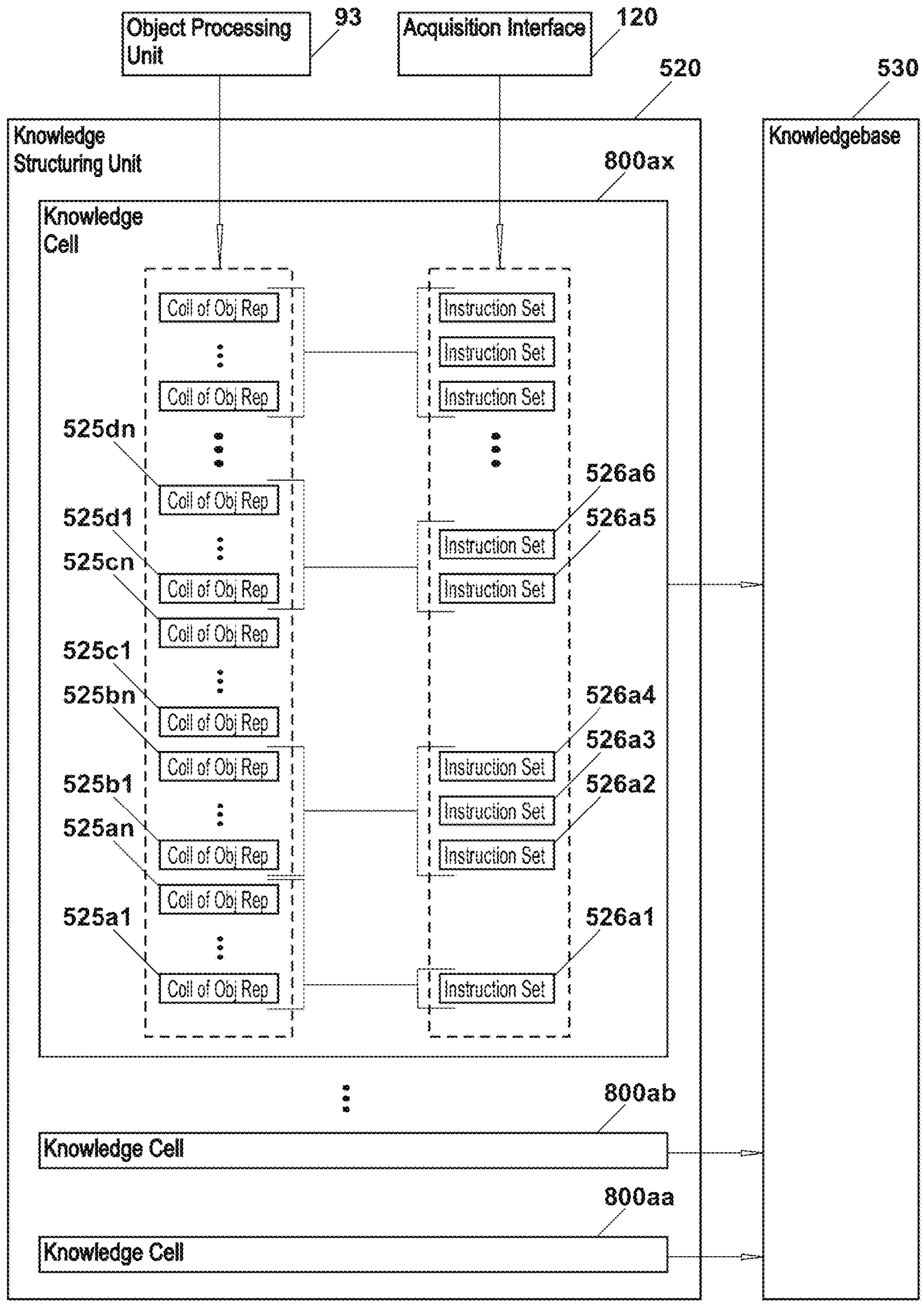
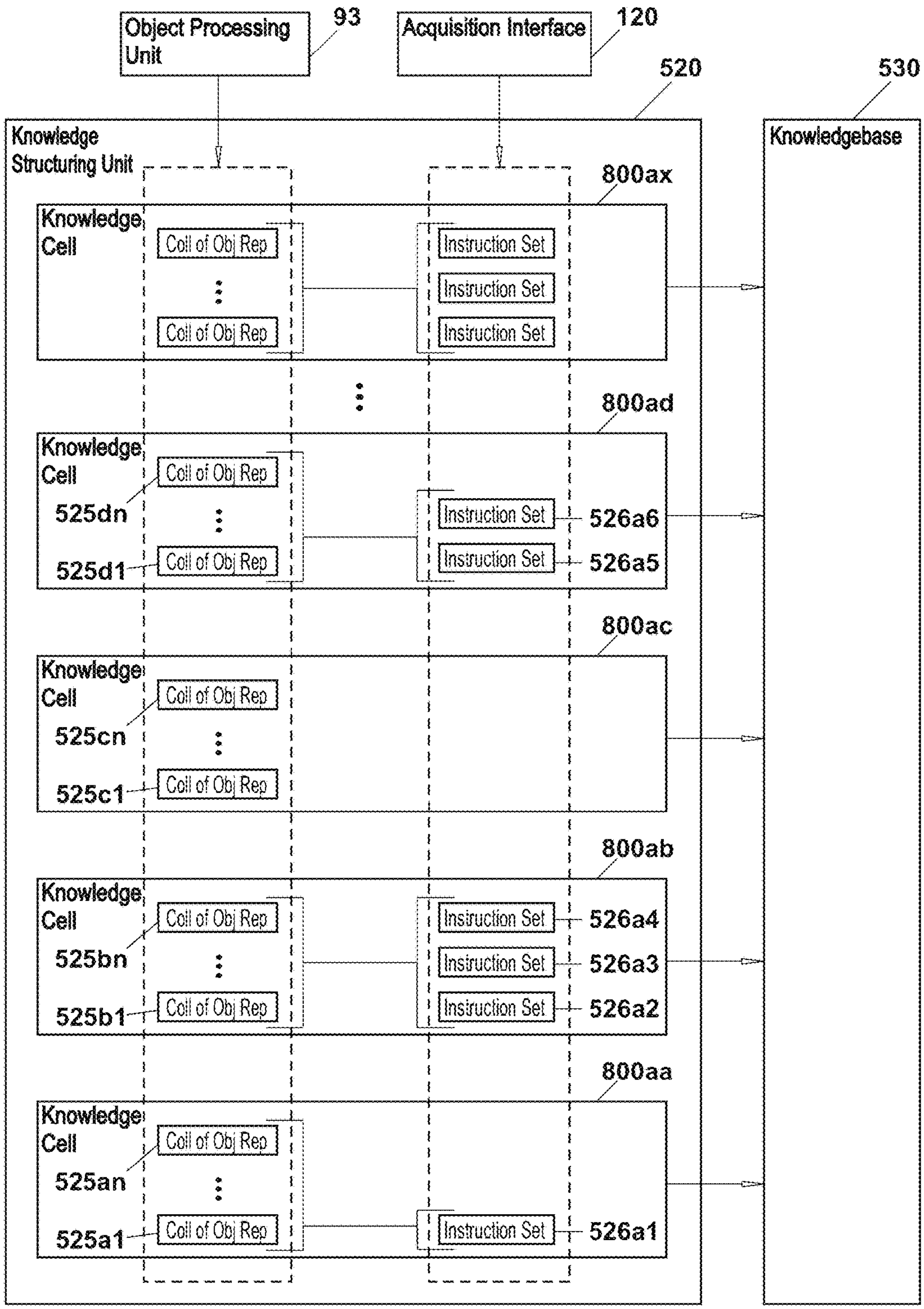

Object Processing Unit
93
Acquisition Interface
120
520
530
Knowledge Structuring Unit
Knowledgebase
800ax
Knowledge Cell
Coll of Obj Rep
Instruction Set
525dn
525d1
525cn
525c1
525bn
525b1
525an
525a1
526a6
526a5
526a4
526a3
526a2
526a1
800ab
Knowledge Cell
800aa
Knowledge Cell

FIG. 15



Object Processing Unit
93
Acquisition Interface
120
520
530
Knowledge Structuring Unit
Knowledgebase
800ax
Knowledge Cell
Coll of Obj Rep
Coll of Obj Rep
Instruction Set
Instruction Set
Instruction Set
800ad
Knowledge Cell
525dn
525d1
Coll of Obj Rep
Coll of Obj Rep
Instruction Set
Instruction Set
526a6
526a5
800ac
Knowledge Cell
525cn
525c1
Coll of Obj Rep
Coll of Obj Rep
800ab
Knowledge Cell
525bn
525b1
Coll of Obj Rep
Coll of Obj Rep
Instruction Set
Instruction Set
Instruction Set
526a4
526a3
526a2
800aa
Knowledge Cell
525an
525a1
Coll of Obj Rep
Coll of Obj Rep
Instruction Set
526a1

FIG. 16

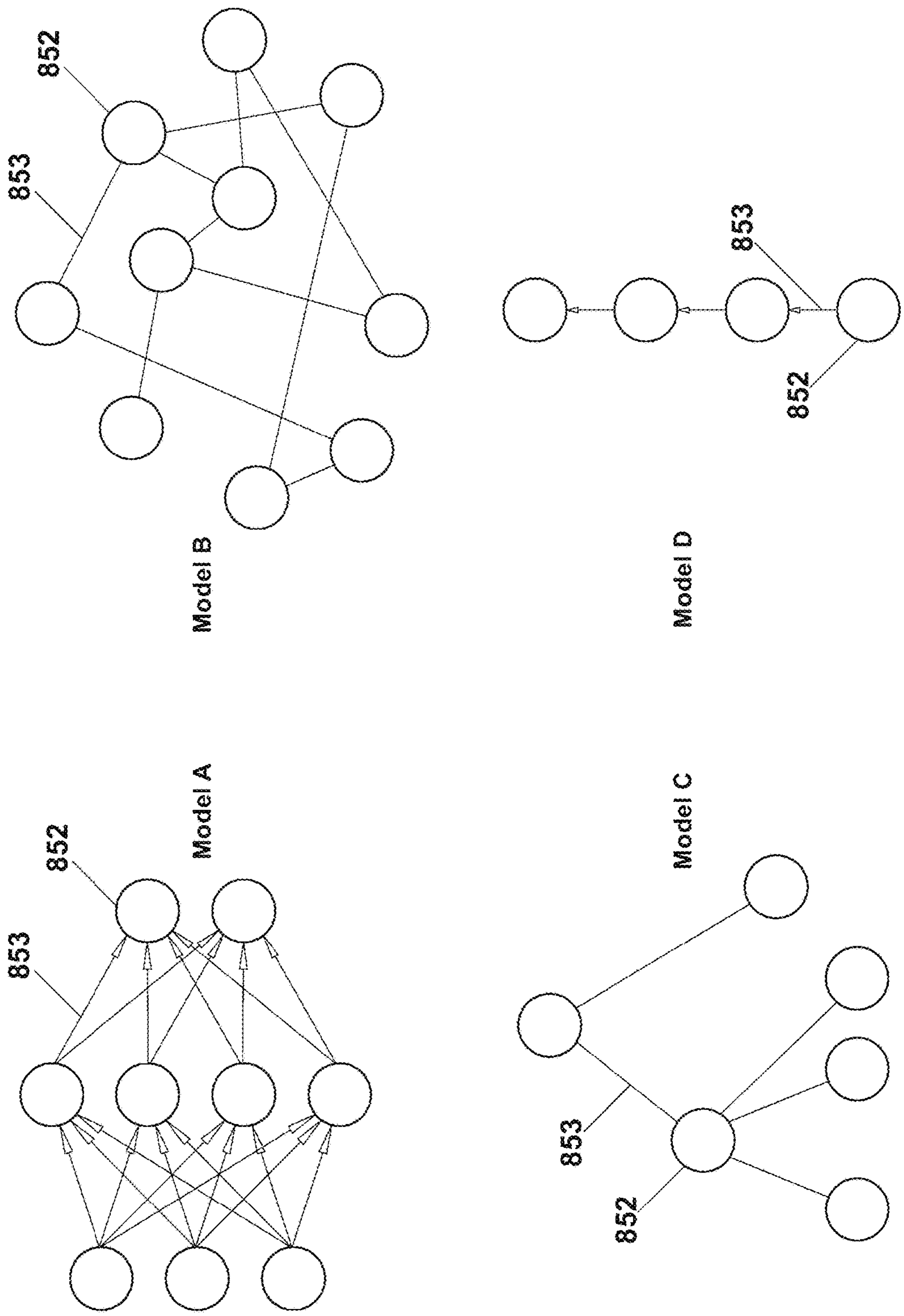


FIG. 17

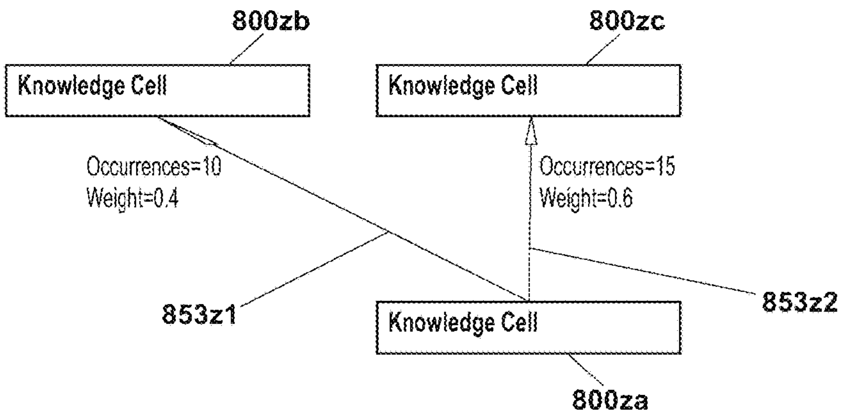


FIG. 18A

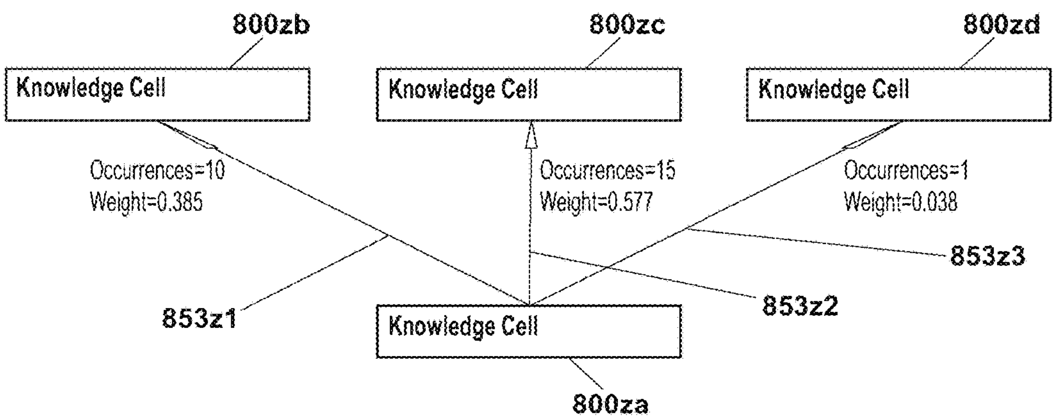


FIG. 18B

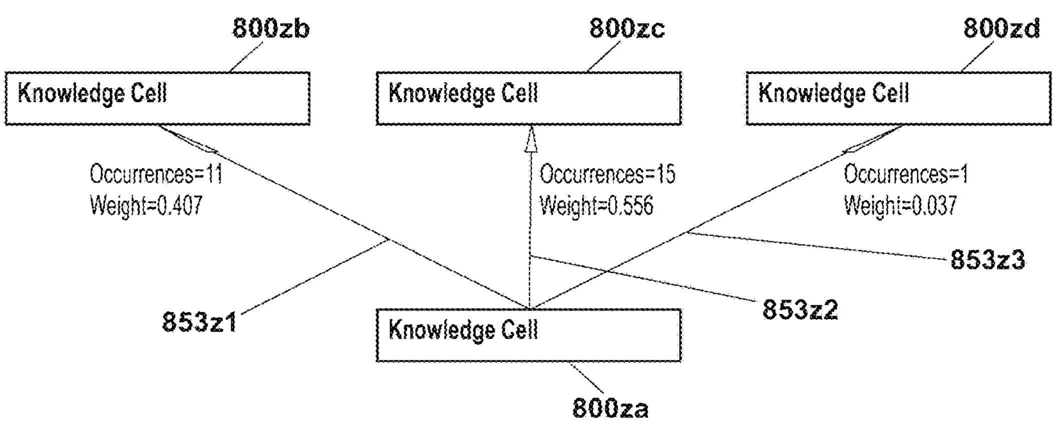


FIG. 18C

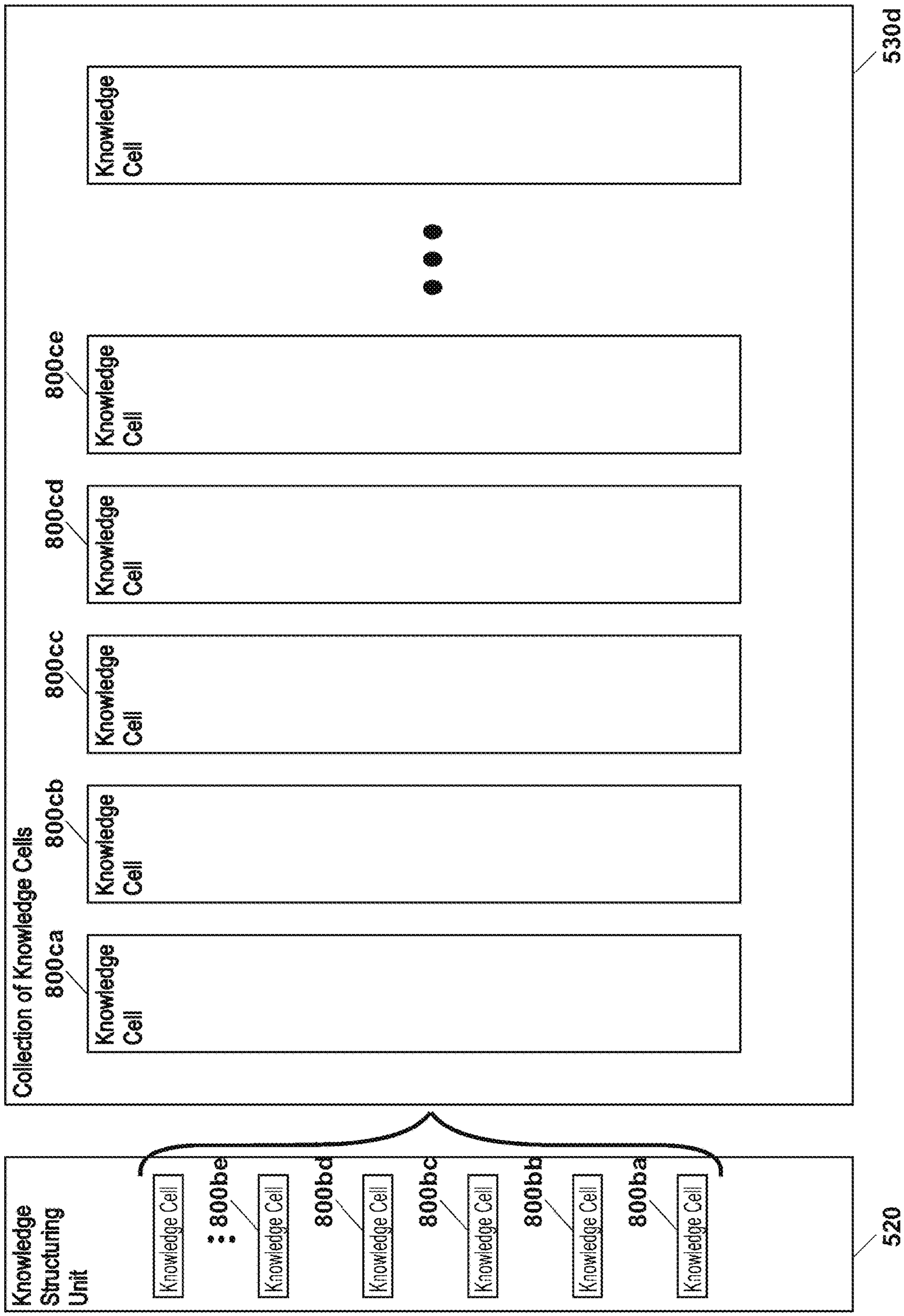

Collection of Knowledge Cells
800ca
800cb
800cc
800cd
800ce
Knowledge Cell
530d
Knowledge Structuring Unit
800ba
800bb
800bc
800bd
800be
520

FIG. 19

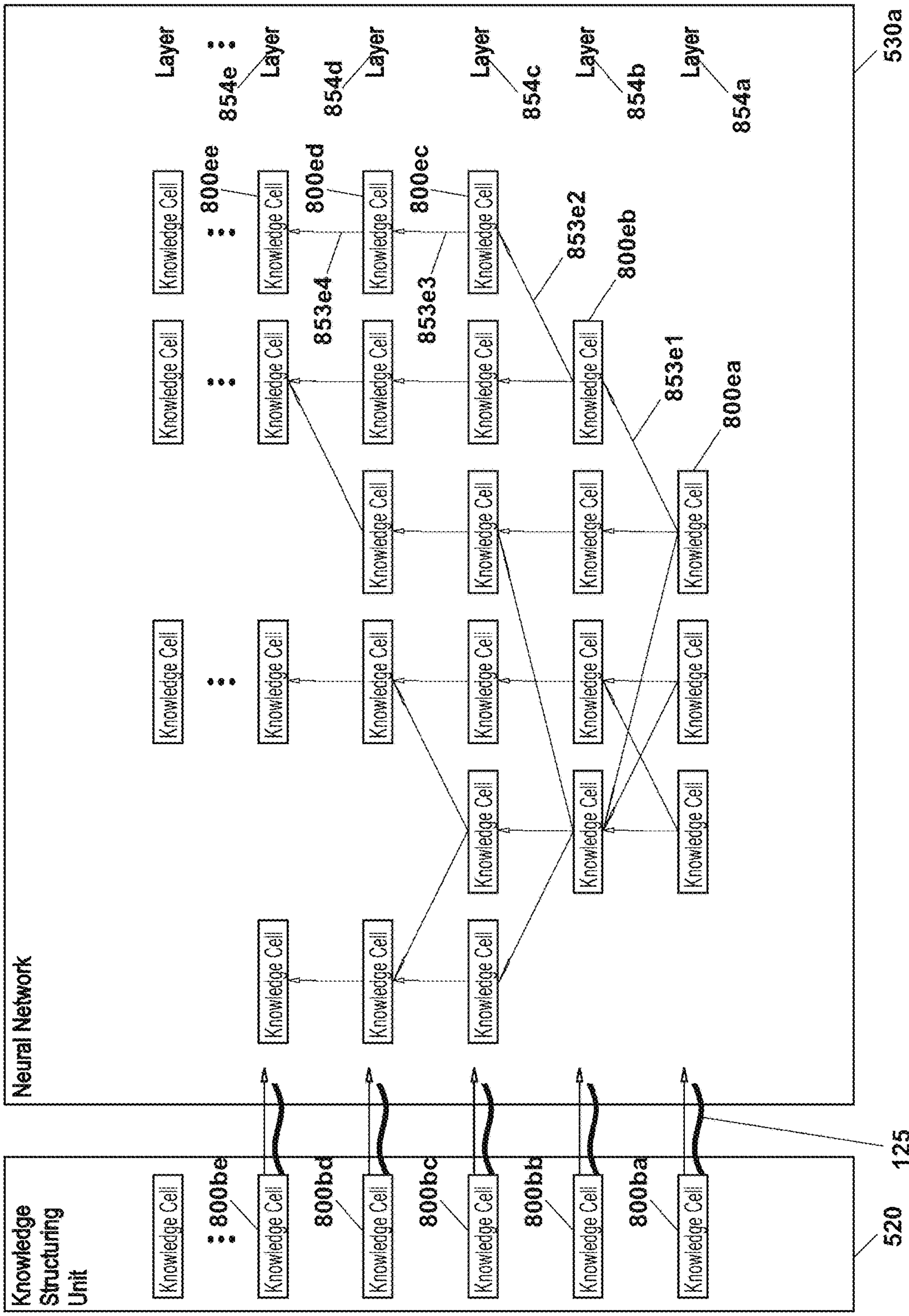

Neural Network
Knowledge Structuring Unit
Knowledge Cell
Layer
854a
854b
854c
854d
854e
800ea
800eb
800ec
800ed
800ee
853e1
853e2
853e3
853e4
800ba
800bb
800bc
800bd
800be
530a
520
125

FIG. 20

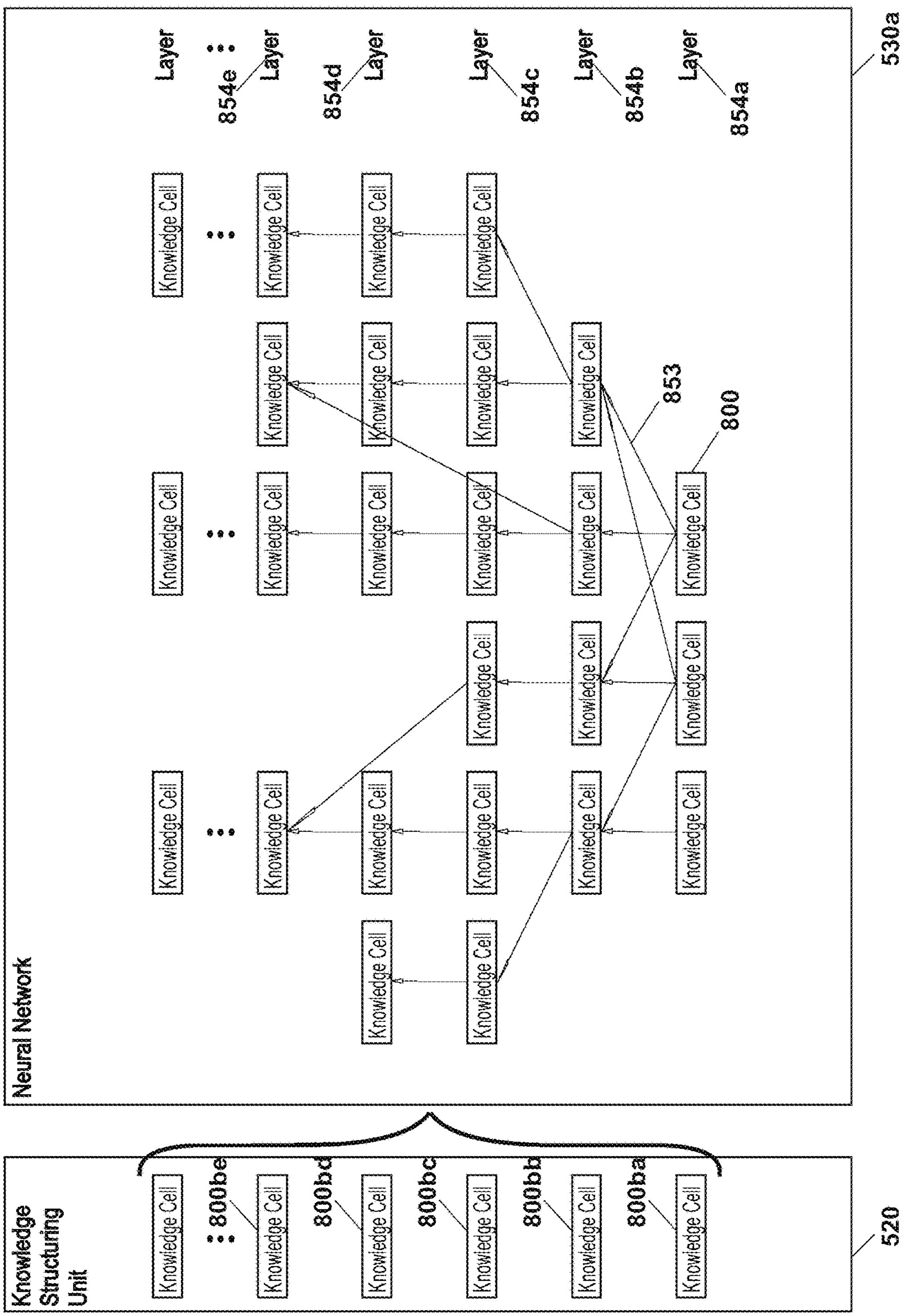

Neural Network
Knowledge Cell
Layer
854a
854b
854c
854d
854e
853
800
530a
Knowledge Structuring Unit
800ba
800bb
800bc
800bd
800be
520

FIG. 21

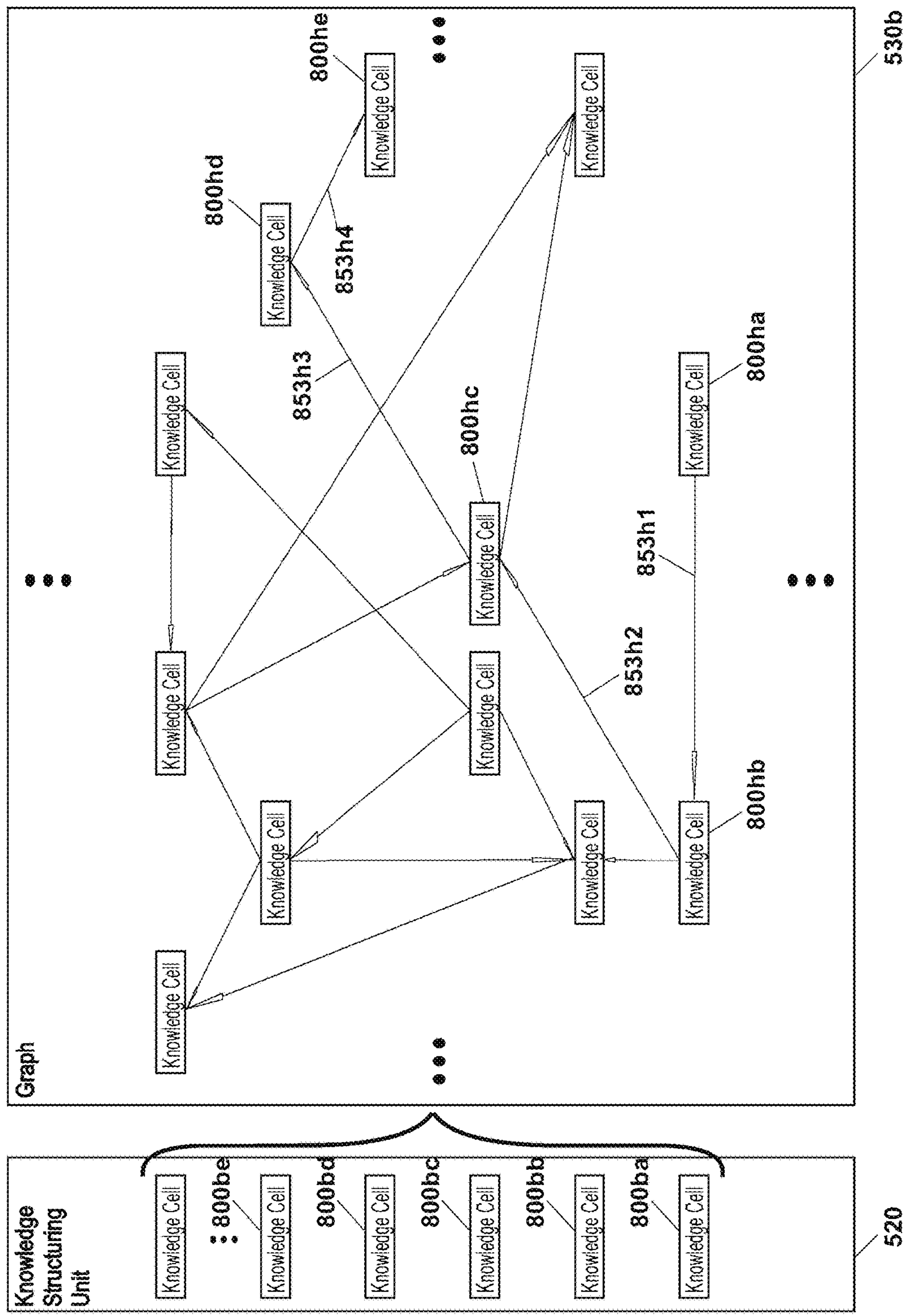

Graph
Knowledge Cell
800ha
800hb
800hc
800hd
800he
853h1
853h2
853h3
853h4
530b
Knowledge Structuring Unit
800ba
800bb
800bc
800bd
800be
520

FIG. 22

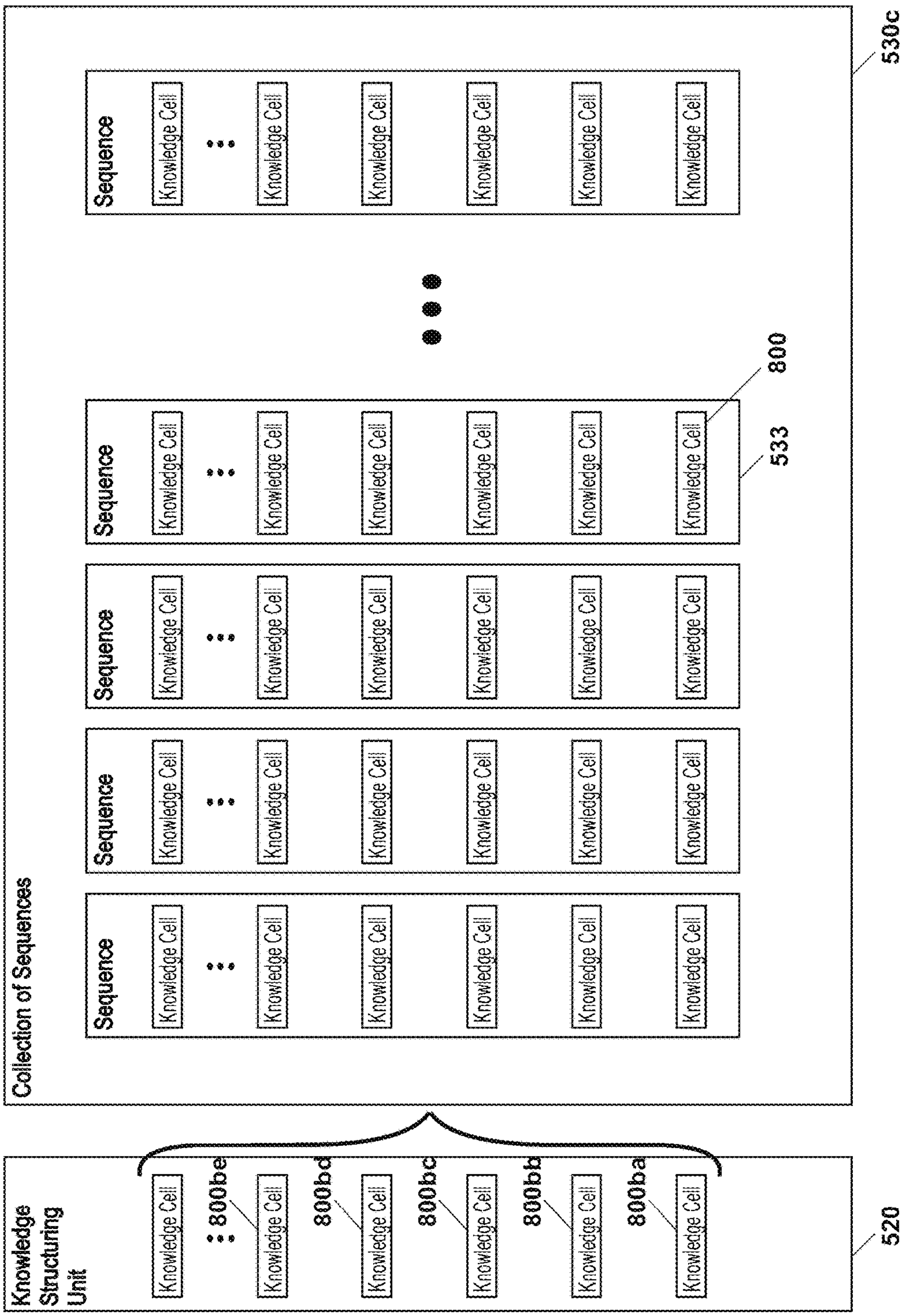

Knowledge Structuring Unit
Knowledge Cell
800be
800bd
800bc
800bb
800ba
520
Collection of Sequences
Sequence
Knowledge Cell
533
800
530c

FIG. 23

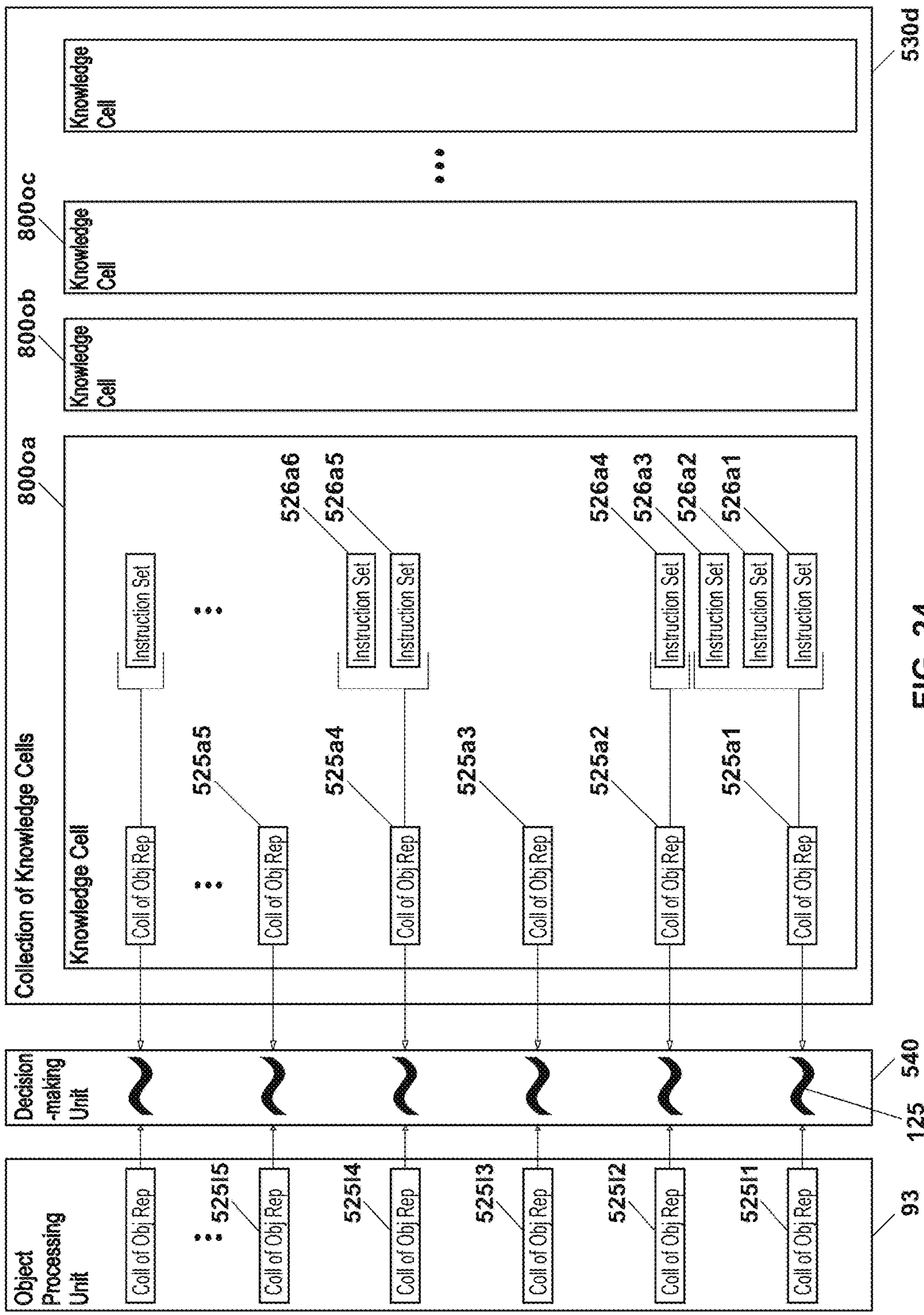

Object Processing Unit
Coll of Obj Rep
525l1
525l2
525l3
525l4
525l5
93
Decision -making Unit
125
540
Collection of Knowledge Cells
Knowledge Cell
Coll of Obj Rep
525a1
525a2
525a3
525a4
525a5
Instruction Set
526a1
526a2
526a3
526a4
526a5
526a6
800oa
800ob
800oc
Knowledge Cell
530d

FIG. 24

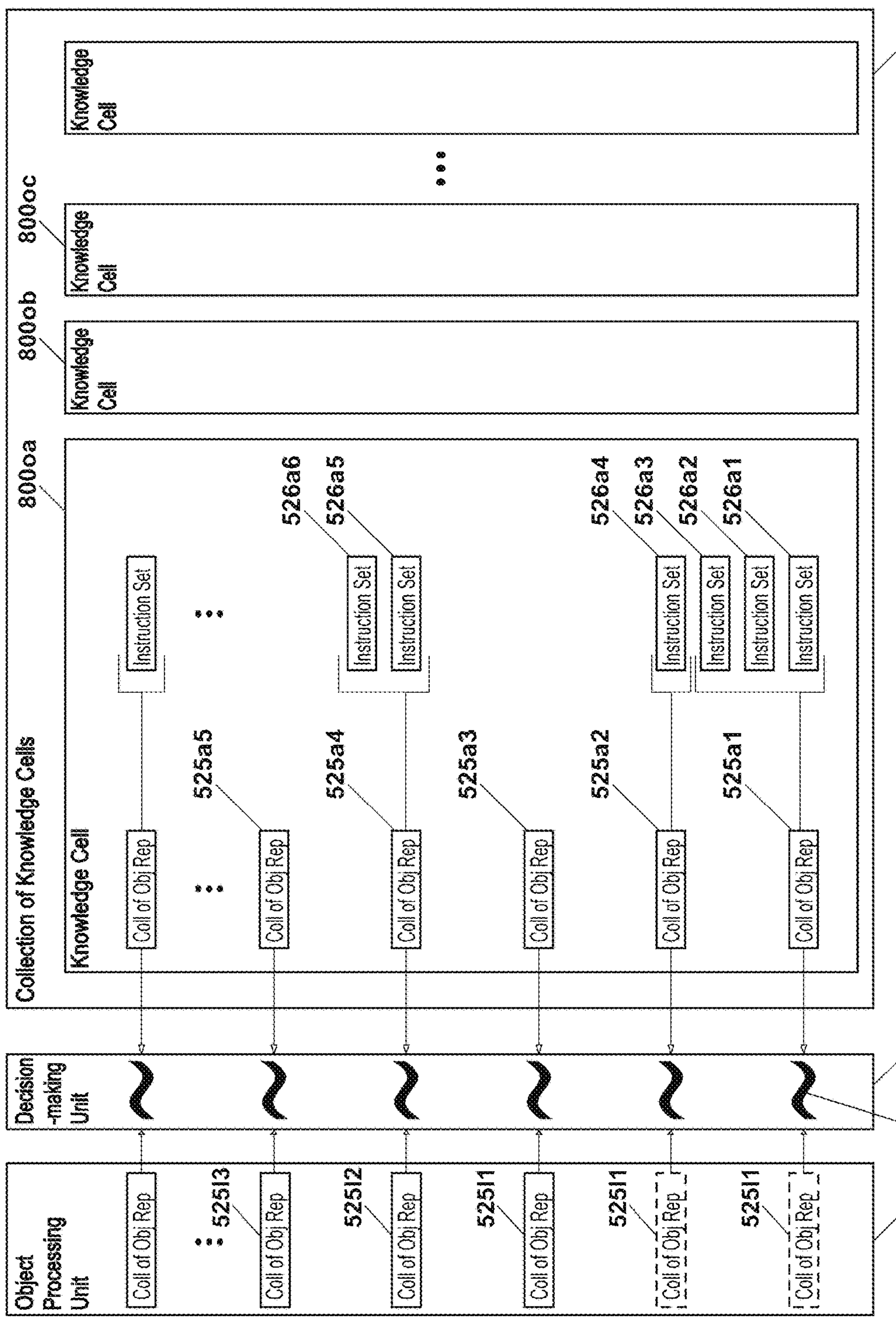

Object Processing Unit
Decision-making Unit
Collection of Knowledge Cells
Knowledge Cell
Coll of Obj Rep
Instruction Set
525l1
525l2
525l3
525a1
525a2
525a3
525a4
525a5
526a1
526a2
526a3
526a4
526a5
526a6
800oa
800ob
800oc
530d
540
125
93

FIG. 25

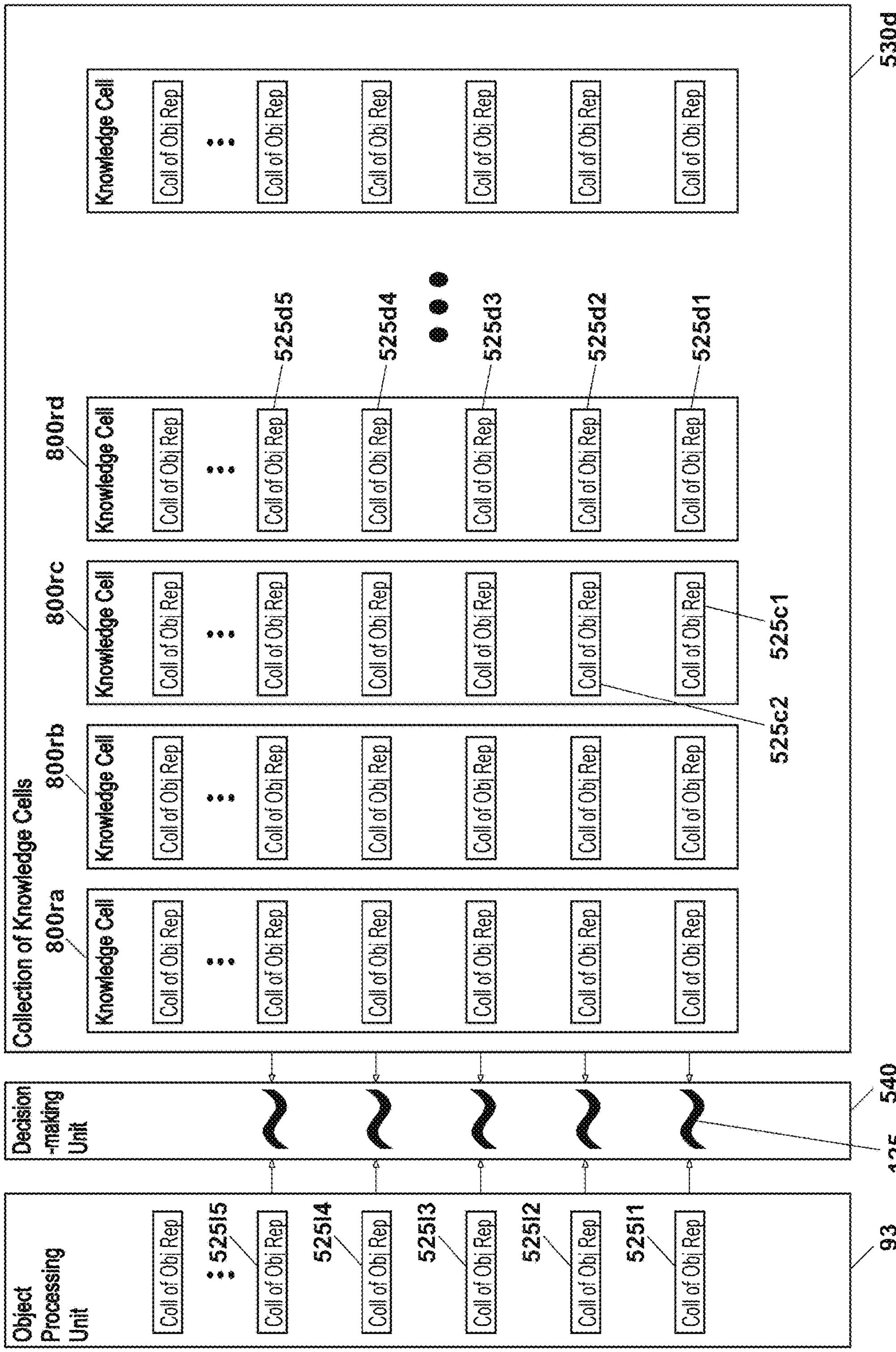

Object Processing Unit
Coll of Obj Rep
52515
52514
52513
52512
52511
93
Decision -making Unit
540
125
Collection of Knowledge Cells
Knowledge Cell
800ra
800rb
800rc
800rd
525c1
525c2
525d1
525d2
525d3
525d4
525d5
530d

FIG. 26

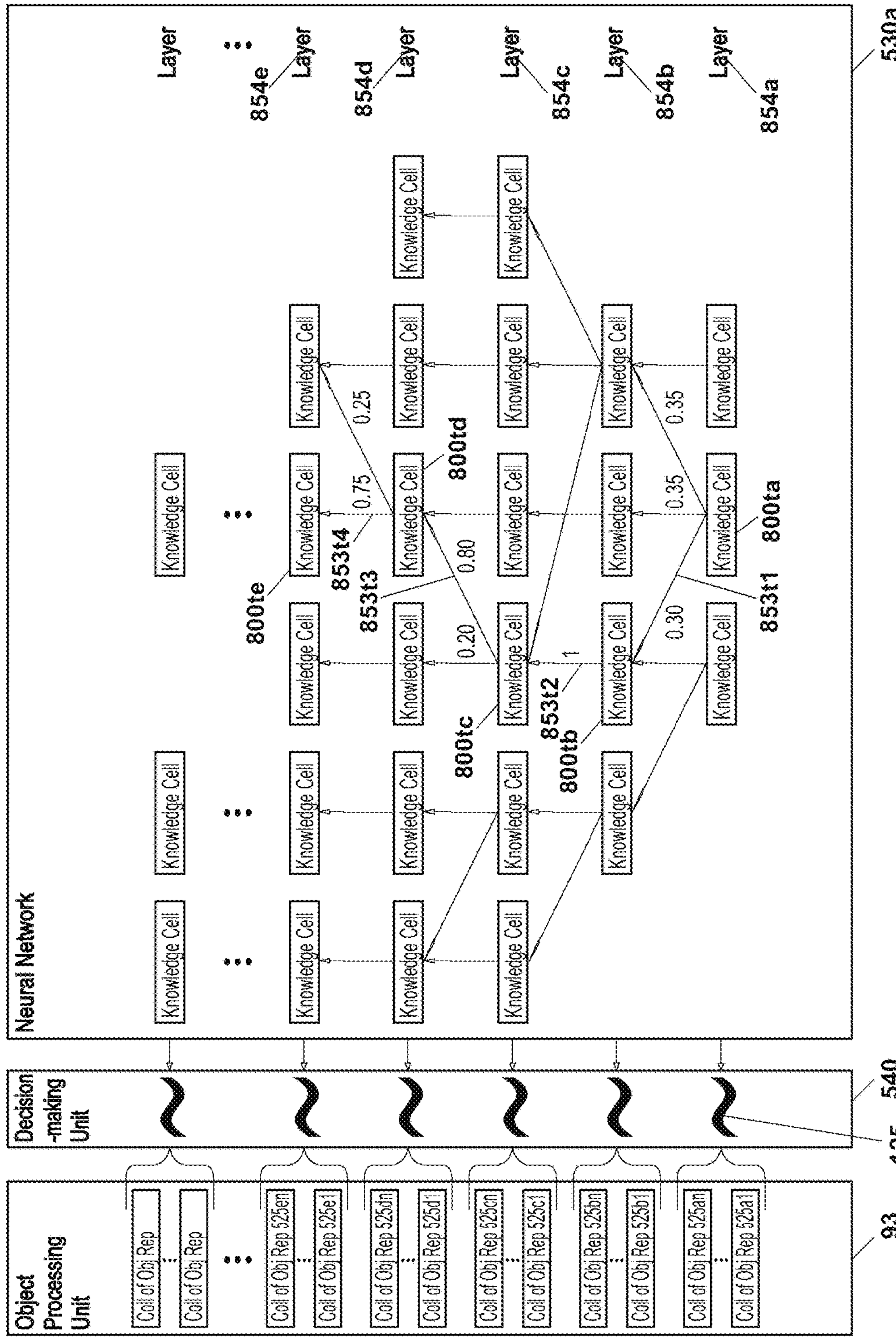

Neural Network
Knowledge Cell
Layer
854a
854b
854c
854d
854e
800ta
800tb
800tc
800td
800te
853t1
853t2
853t3
853t4
0.25
0.75
0.80
0.20
1
0.35
0.35
0.30
530a
Decision -making Unit
540
125
Object Processing Unit
Coll of Obj Rep 525a1
Coll of Obj Rep 525an
Coll of Obj Rep 525b1
Coll of Obj Rep 525bn
Coll of Obj Rep 525c1
Coll of Obj Rep 525cn
Coll of Obj Rep 525d1
Coll of Obj Rep 525dn
Coll of Obj Rep 525e1
Coll of Obj Rep 525en
Coll of Obj Rep
93

FIG. 27

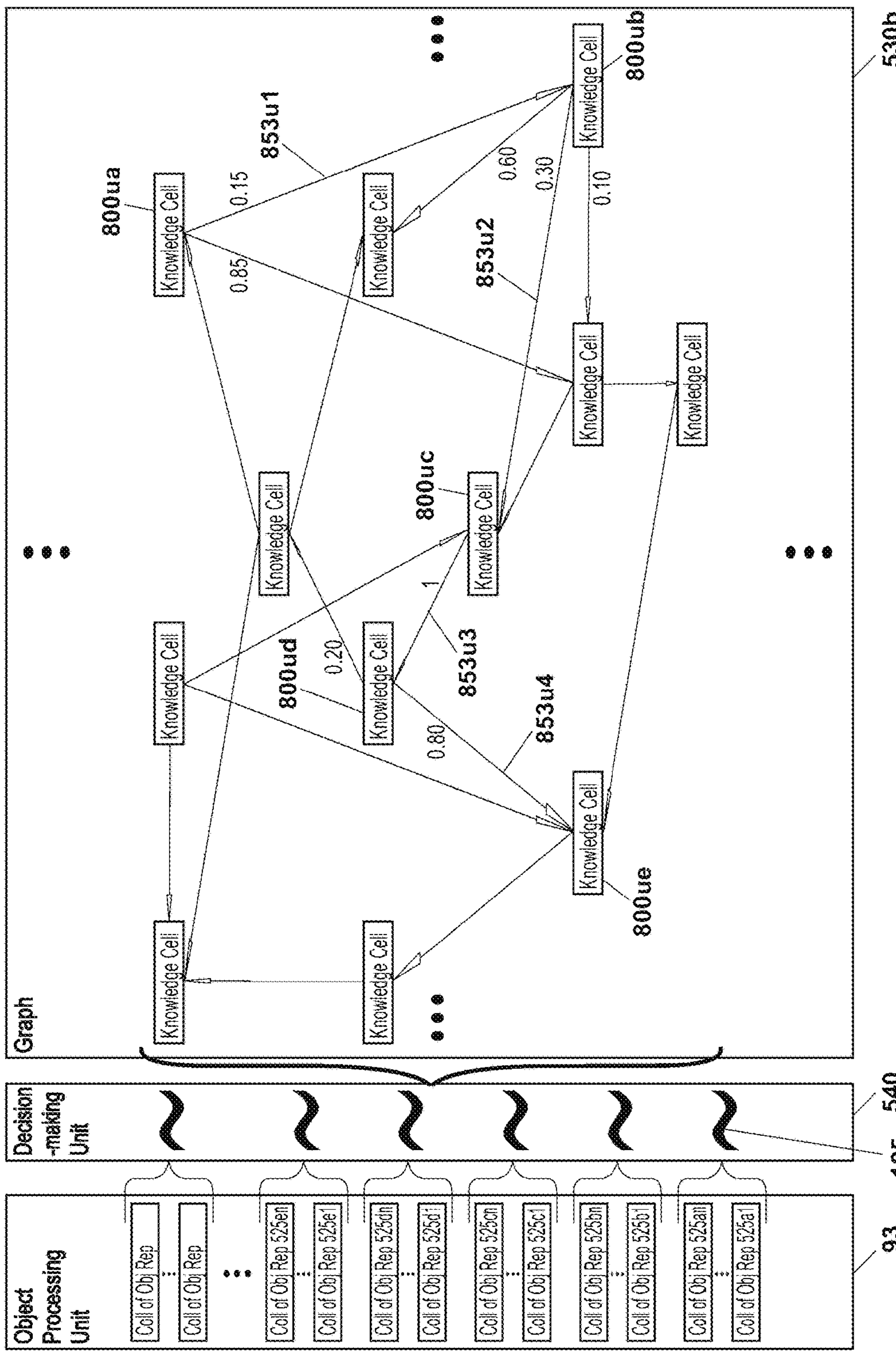
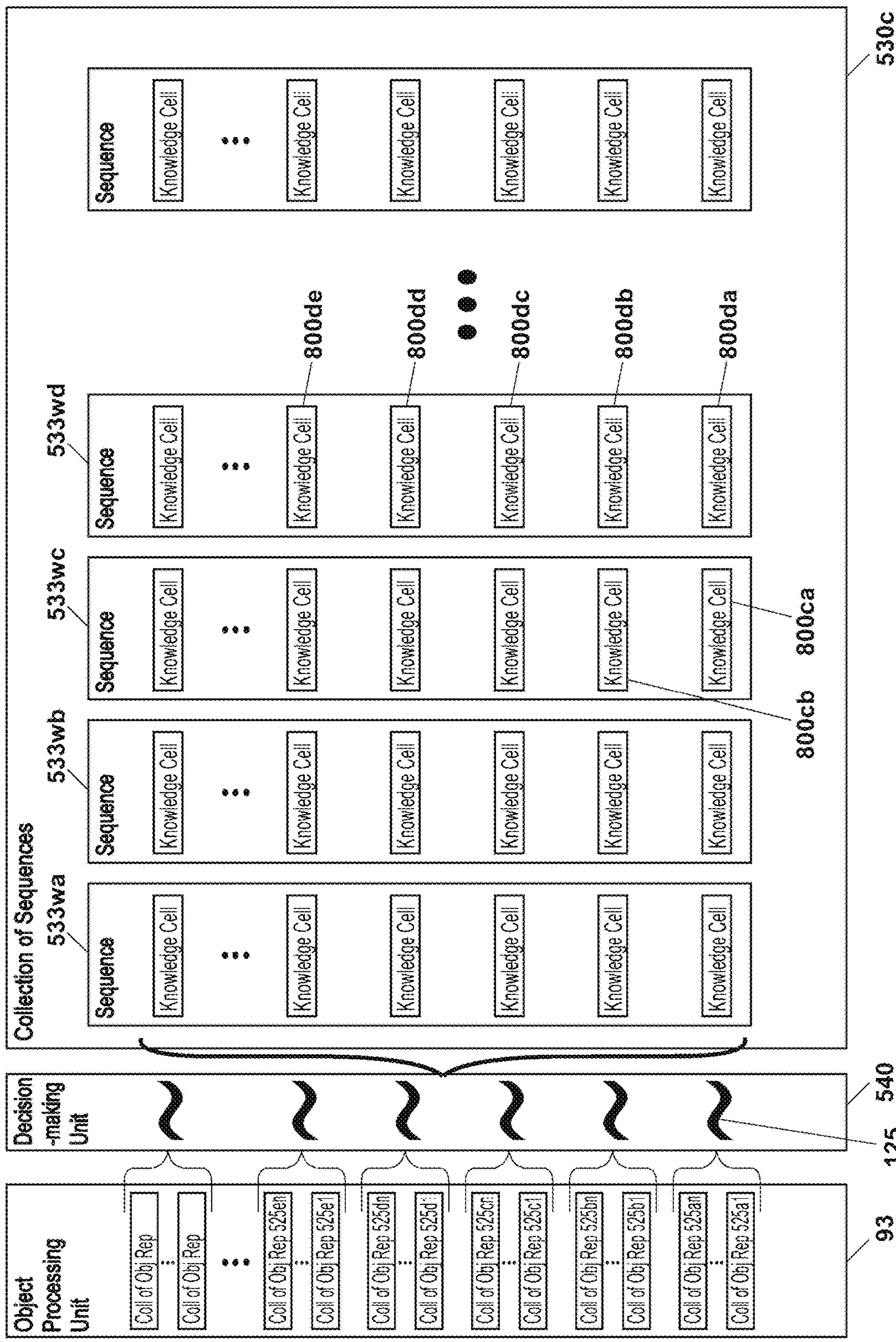

Graph
Knowledge Cell
800ua
800ub
800uc
800ud
800ue
853u1
853u2
853u3
853u4
0.15
0.85
0.60
0.30
0.10
1
0.20
0.80
530b
Decision
-making
Unit
540
125
Object
Processing
Unit
Coll of Obj Rep
Coll of Obj Rep 525en
Coll of Obj Rep 525e1
Coll of Obj Rep 525dn
Coll of Obj Rep 525d1
Coll of Obj Rep 525cn
Coll of Obj Rep 525c1
Coll of Obj Rep 525bn
Coll of Obj Rep 525b1
Coll of Obj Rep 525an
Coll of Obj Rep 525a1
93

FIG. 28



Object Processing Unit
Coll of Obj Rep
Coll of Obj Rep 525en
Coll of Obj Rep 525e1
Coll of Obj Rep 525dn
Coll of Obj Rep 525d1
Coll of Obj Rep 525cn
Coll of Obj Rep 525c1
Coll of Obj Rep 525bn
Coll of Obj Rep 525b1
Coll of Obj Rep 525an
Coll of Obj Rep 525a1
93
Decision -making Unit
540
125
Collection of Sequences
Sequence
Knowledge Cell
533wa
533wb
533wc
533wd
800cb
800ca
800da
800db
800dc
800dd
800de
530c

FIG. 29

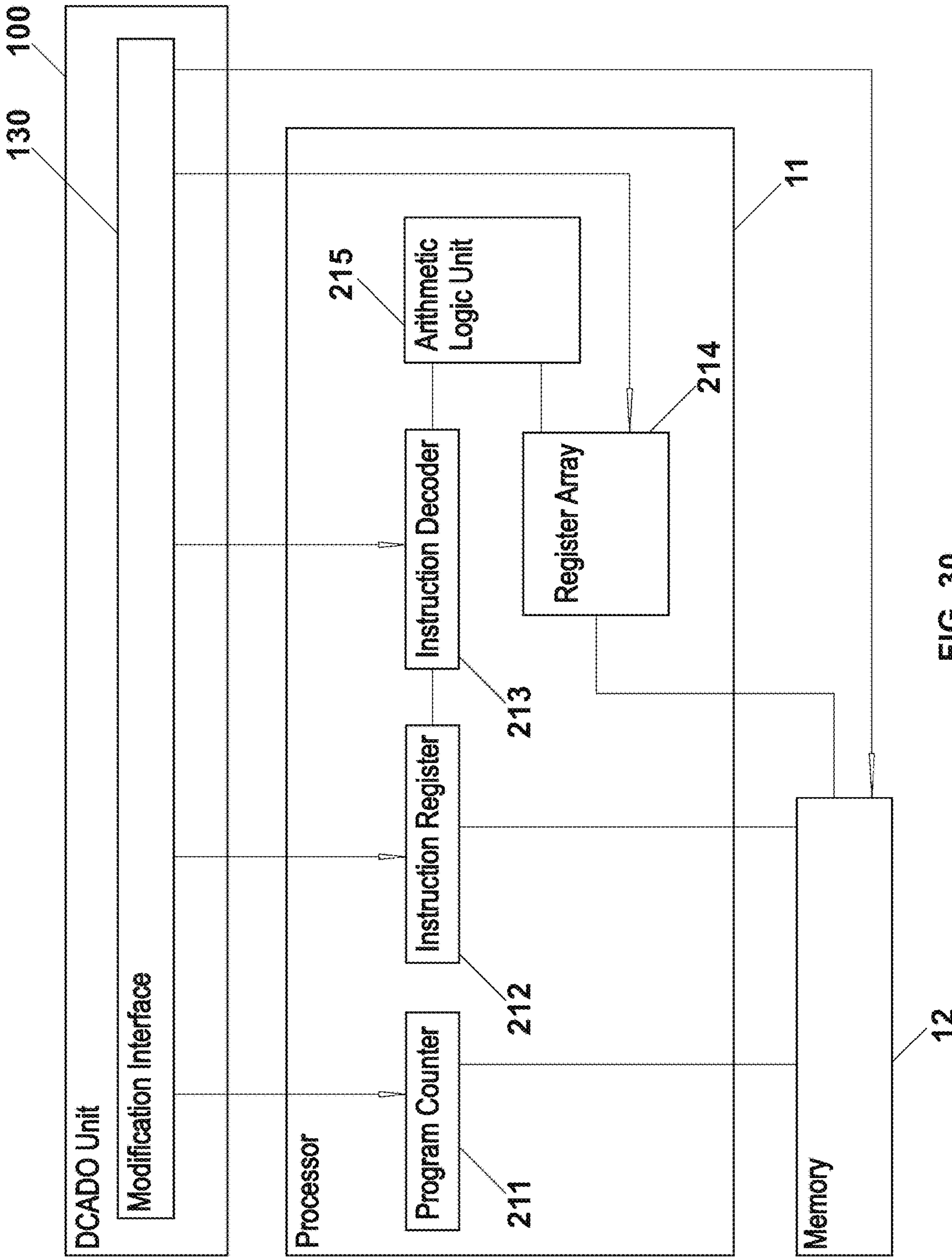

DCADO Unit
Modification Interface
Processor
Program Counter
Instruction Register
Instruction Decoder
Arithmetic Logic Unit
Register Array
Memory
100
130
11
12
211
212
213
214
215

FIG. 30

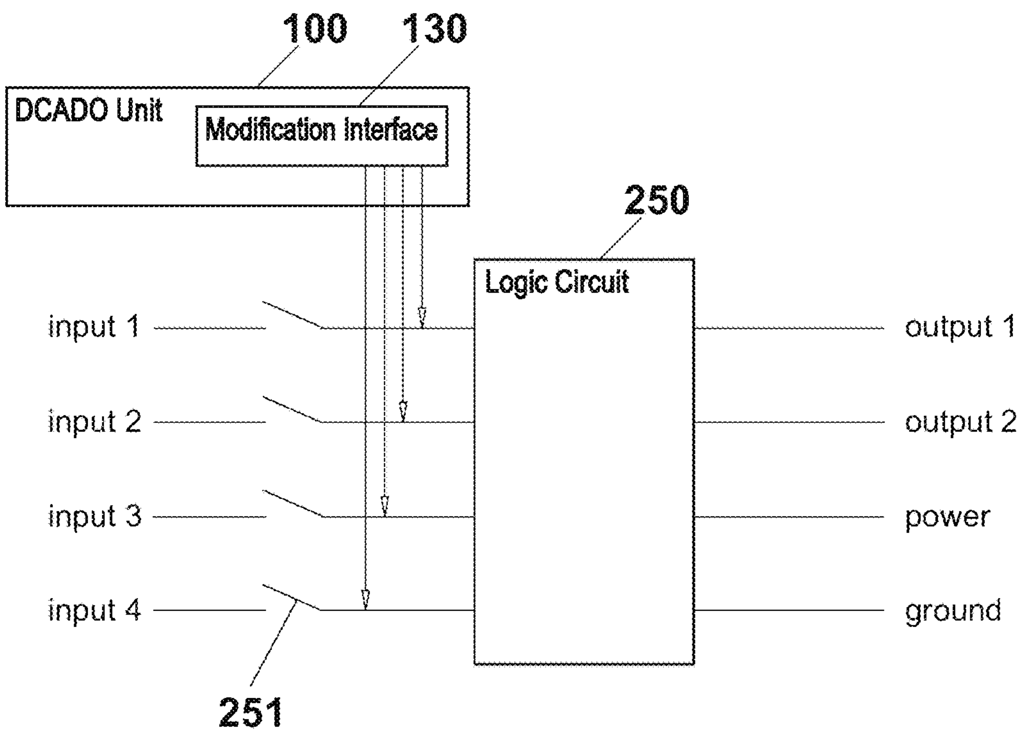


FIG. 31A

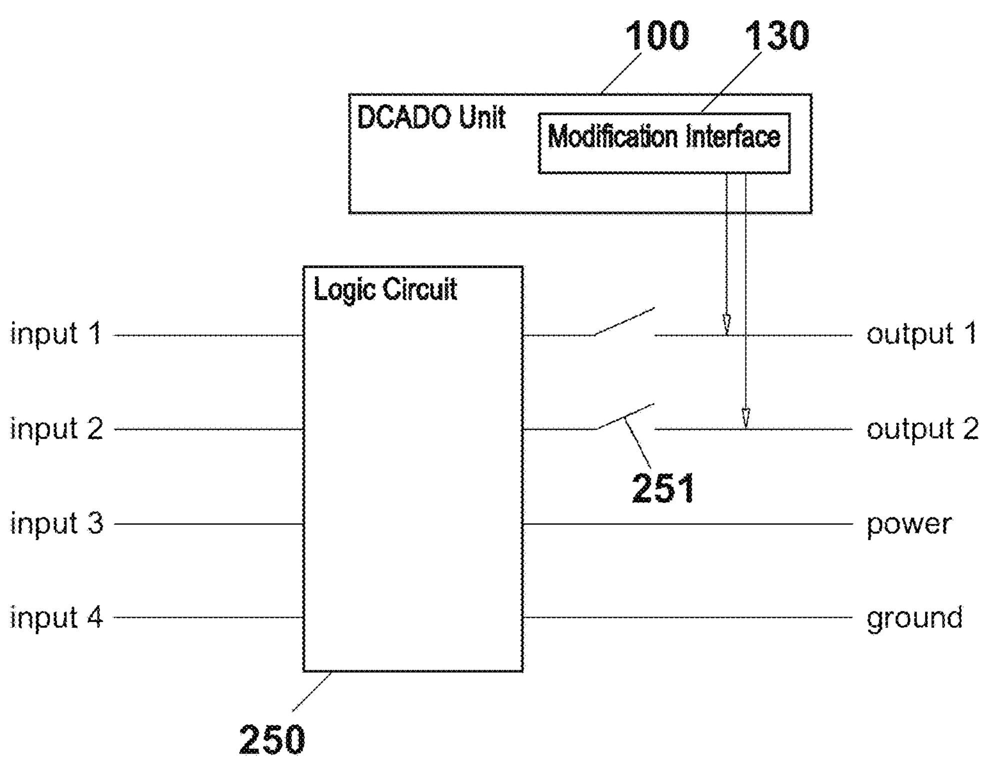


FIG. 31B

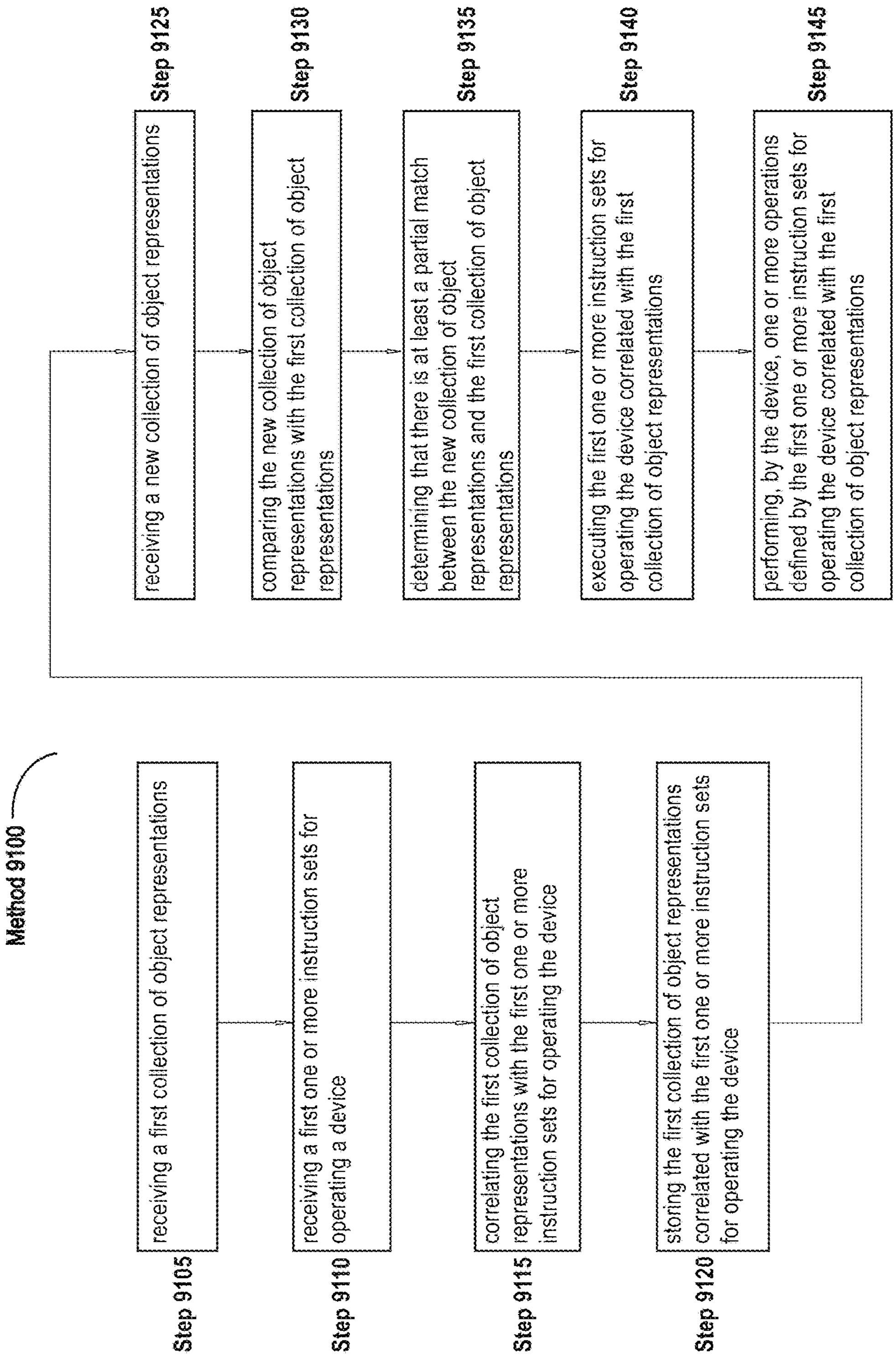

Method 9100
Step 9105
receiving a first collection of object representations
Step 9110
receiving a first one or more instruction sets for operating a device
Step 9115
correlating the first collection of object representations with the first one or more instruction sets for operating the device
Step 9120
storing the first collection of object representations correlated with the first one or more instruction sets for operating the device
Step 9125
receiving a new collection of object representations
Step 9130
comparing the new collection of object representations with the first collection of object representations
Step 9135
determining that there is at least a partial match between the new collection of object representations and the first collection of object representations
Step 9140
executing the first one or more instruction sets for operating the device correlated with the first collection of object representations
Step 9145
performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations

FIG. 32

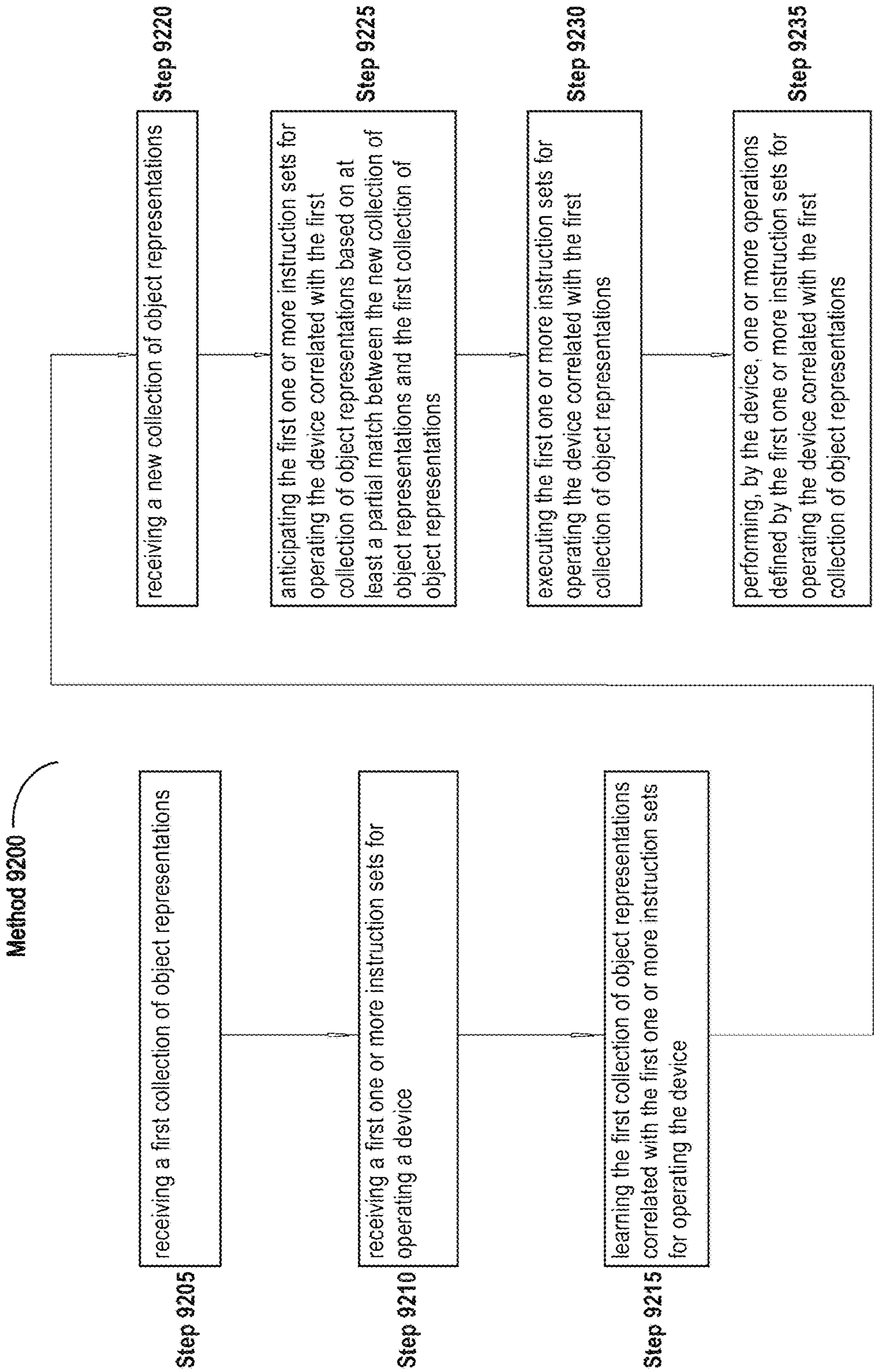

Method 9200
Step 9205
receiving a first collection of object representations
Step 9210
receiving a first one or more instruction sets for operating a device
Step 9215
learning the first collection of object representations correlated with the first one or more instruction sets for operating the device
Step 9220
receiving a new collection of object representations
Step 9225
anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations
Step 9230
executing the first one or more instruction sets for operating the device correlated with the first collection of object representations
Step 9235
performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations

FIG. 33

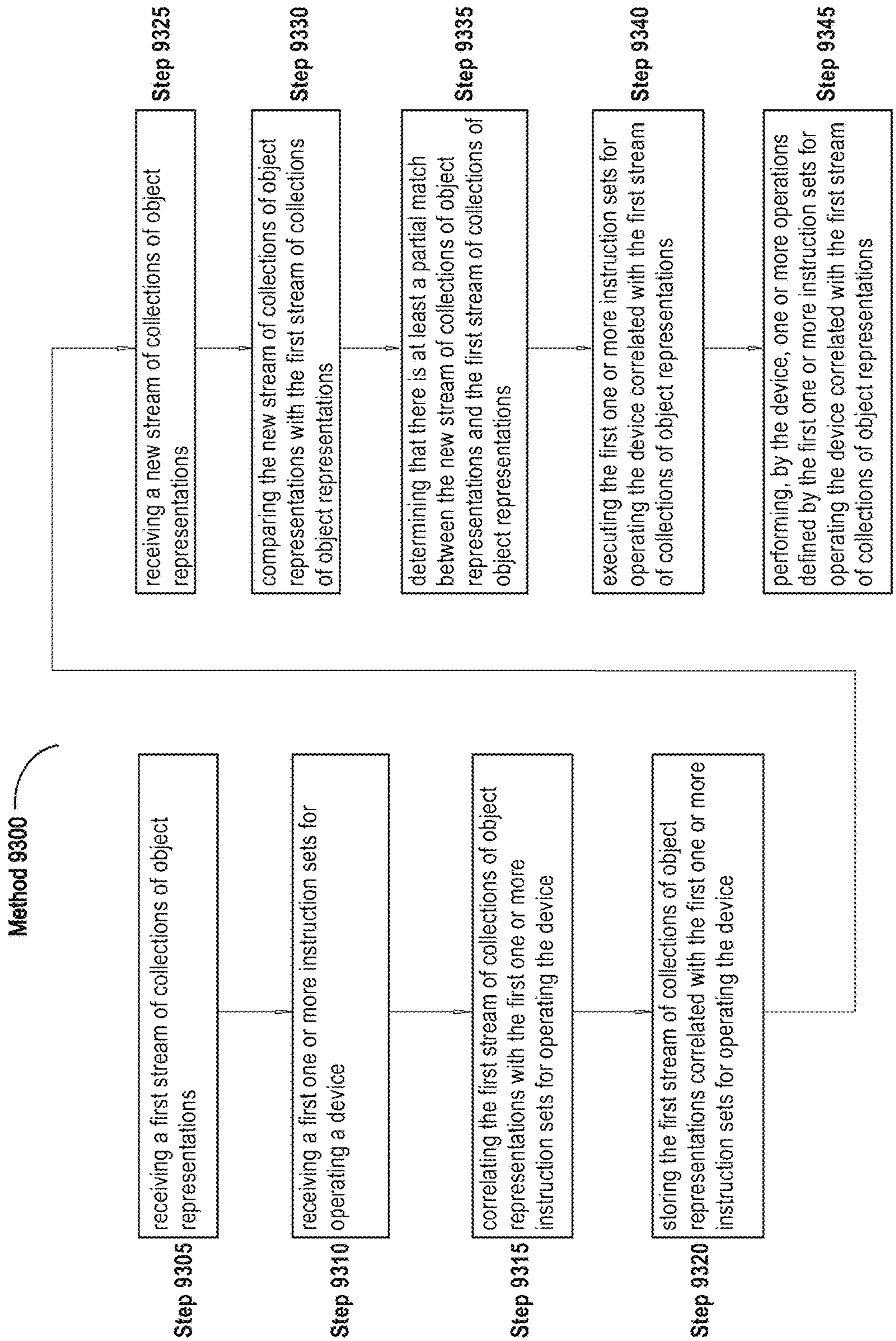

Method 9300
Step 9305
receiving a first stream of collections of object representations
Step 9310
receiving a first one or more instruction sets for operating a device
Step 9315
correlating the first stream of collections of object representations with the first one or more instruction sets for operating the device
Step 9320
storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device
Step 9325
receiving a new stream of collections of object representations
Step 9330
comparing the new stream of collections of object representations with the first stream of collections of object representations
Step 9335
determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations
Step 9340
executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations
Step 9345
performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations

FIG. 34

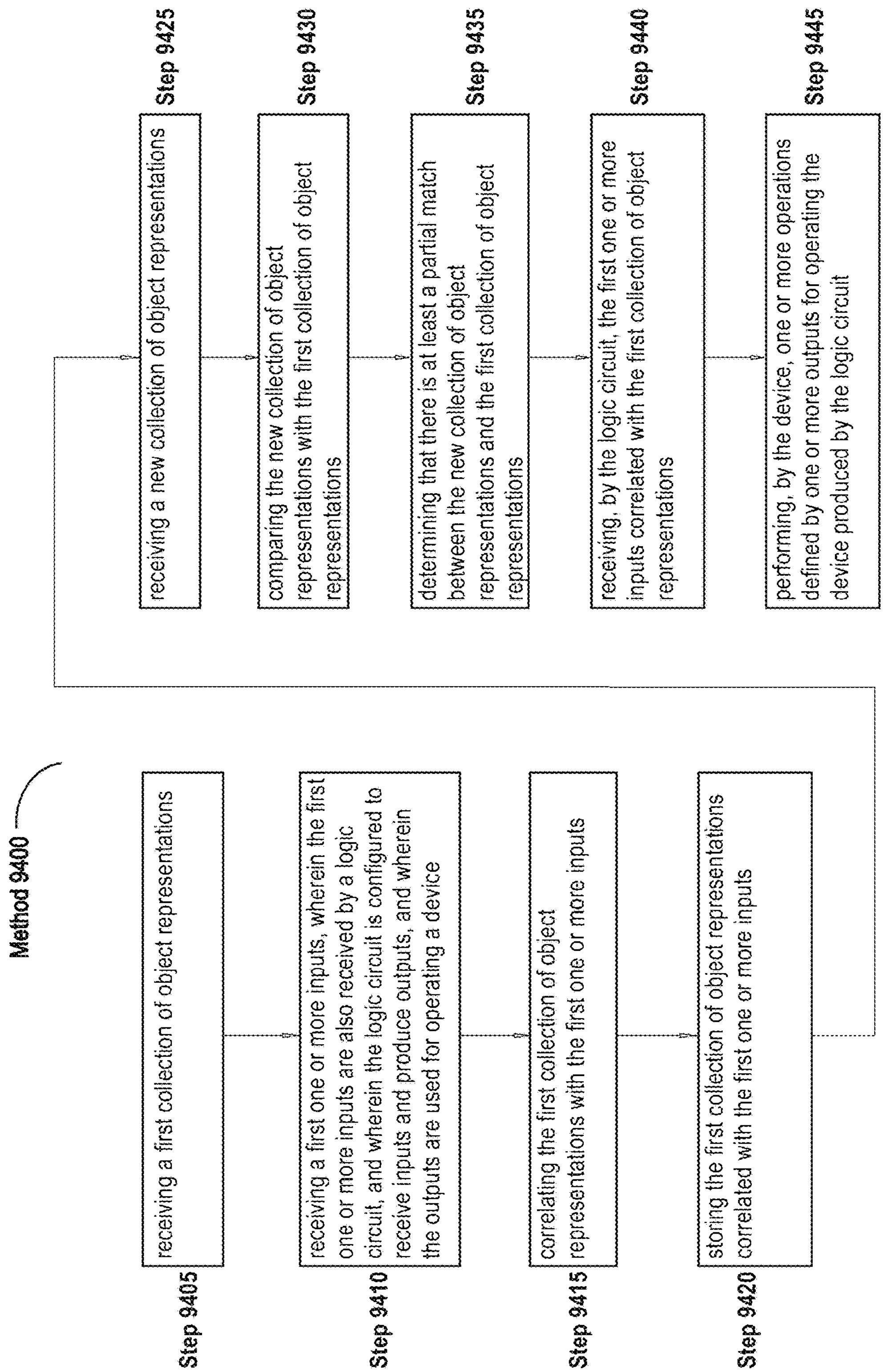

Method 9400
Step 9405
receiving a first collection of object representations
Step 9410
receiving a first one or more inputs, wherein the first one or more inputs are also received by a logic circuit, and wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device
Step 9415
correlating the first collection of object representations with the first one or more inputs
Step 9420
storing the first collection of object representations correlated with the first one or more inputs
Step 9425
receiving a new collection of object representations
Step 9430
comparing the new collection of object representations with the first collection of object representations
Step 9435
determining that there is at least a partial match between the new collection of object representations and the first collection of object representations
Step 9440
receiving, by the logic circuit, the first one or more inputs correlated with the first collection of object representations
Step 9445
performing, by the device, one or more operations defined by one or more outputs for operating the device produced by the logic circuit

FIG. 35

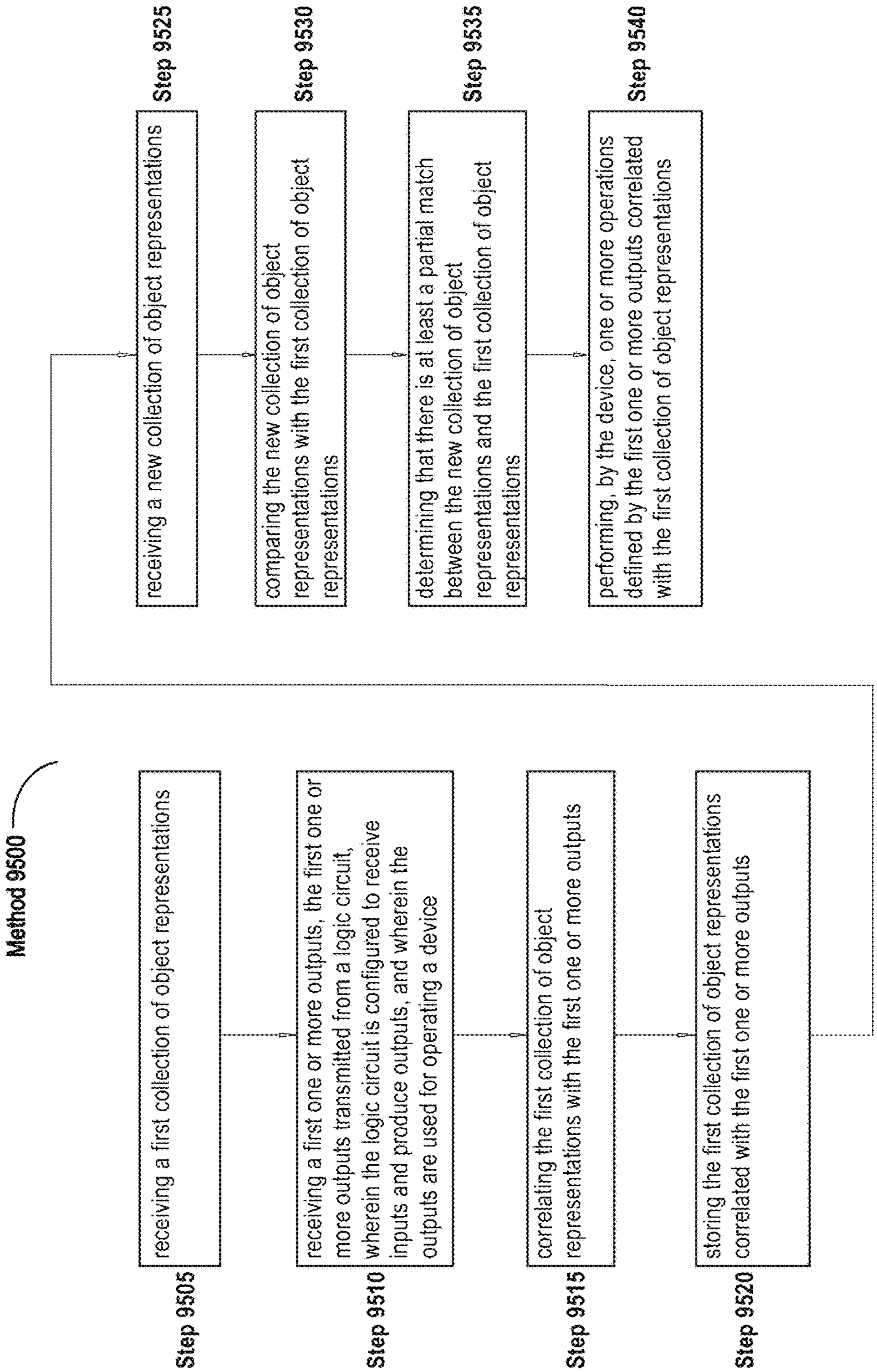

Method 9500
Step 9505
receiving a first collection of object representations
Step 9510
receiving a first one or more outputs, the first one or more outputs transmitted from a logic circuit, wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device
Step 9515
correlating the first collection of object representations with the first one or more outputs
Step 9520
storing the first collection of object representations correlated with the first one or more outputs
Step 9525
receiving a new collection of object representations
Step 9530
comparing the new collection of object representations with the first collection of object representations
Step 9535
determining that there is at least a partial match between the new collection of object representations and the first collection of object representations
Step 9540
performing, by the device, one or more operations defined by the first one or more outputs correlated with the first collection of object representations

FIG. 36

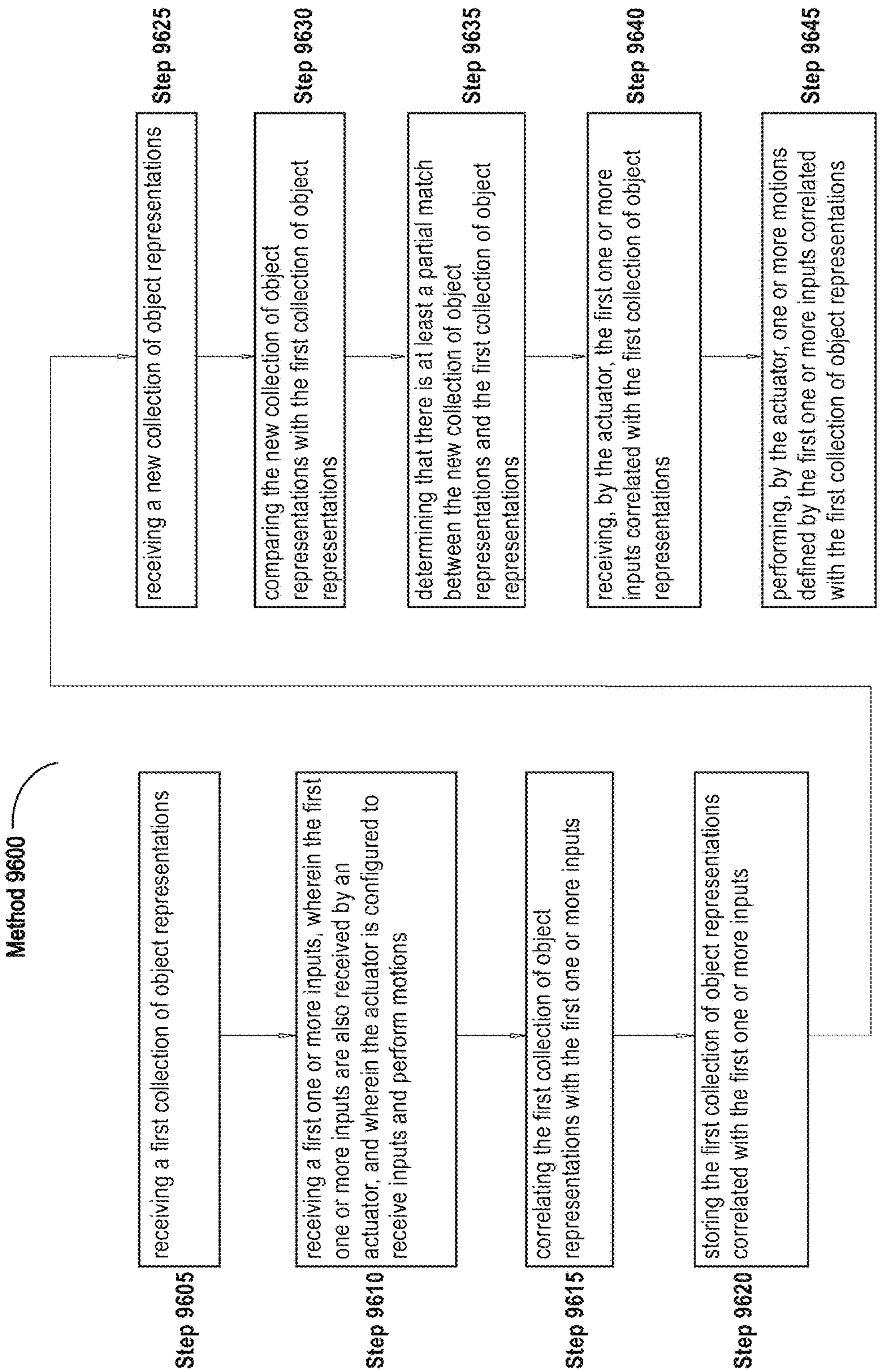

Method 9600
Step 9605
receiving a first collection of object representations
Step 9610
receiving a first one or more inputs, wherein the first one or more inputs are also received by an actuator, and wherein the actuator is configured to receive inputs and perform motions
Step 9615
correlating the first collection of object representations with the first one or more inputs
Step 9620
storing the first collection of object representations correlated with the first one or more inputs
Step 9625
receiving a new collection of object representations
Step 9630
comparing the new collection of object representations with the first collection of object representations
Step 9635
determining that there is at least a partial match between the new collection of object representations and the first collection of object representations
Step 9640
receiving, by the actuator, the first one or more inputs correlated with the first collection of object representations
Step 9645
performing, by the actuator, one or more motions defined by the first one or more inputs correlated with the first collection of object representations

FIG. 37

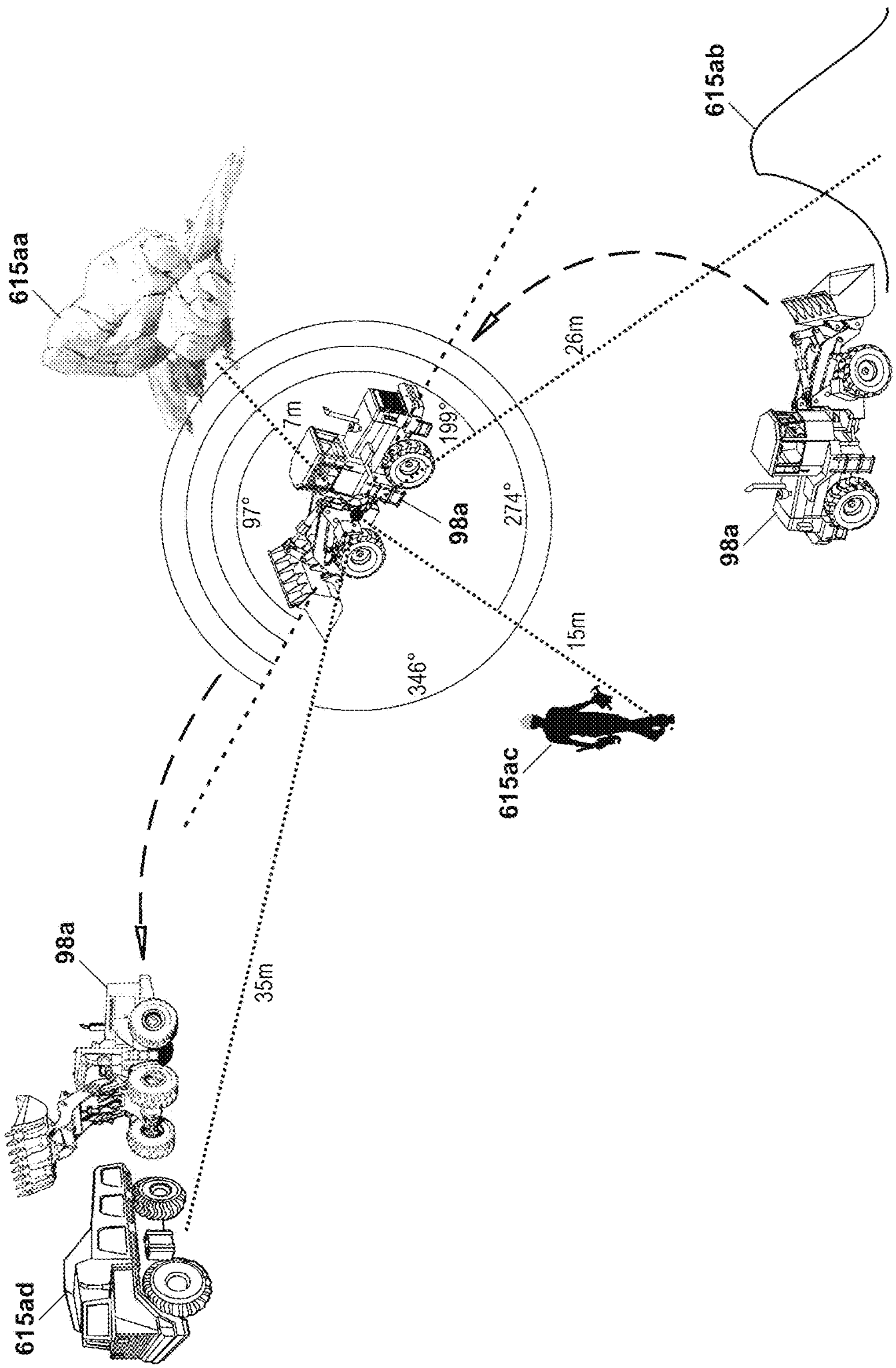

615aa
615ab
615ac
615ad
98a
98a
98a
7m
26m
15m
35m
97°
199°
274°
346°

FIG. 38

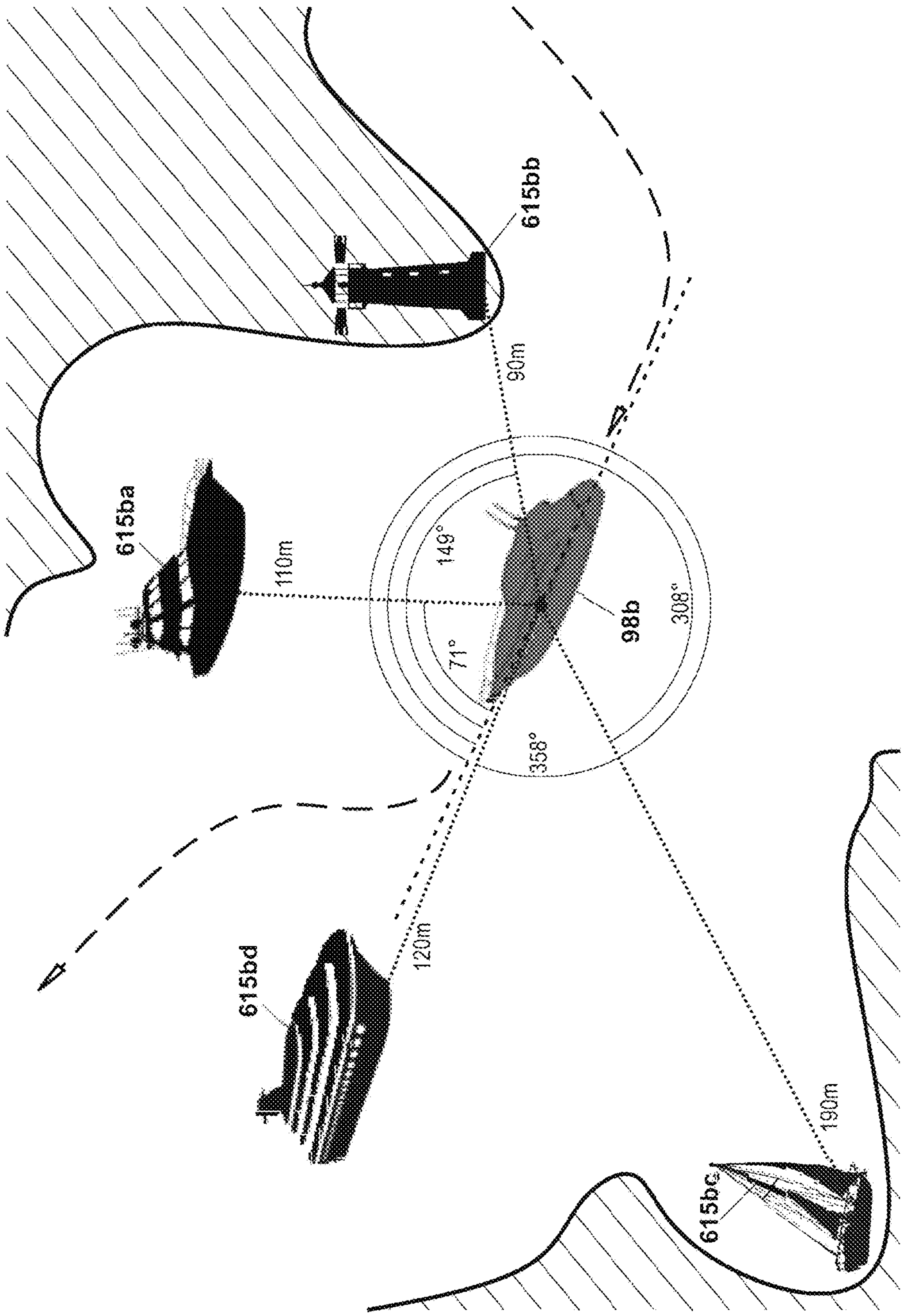


FIG. 39

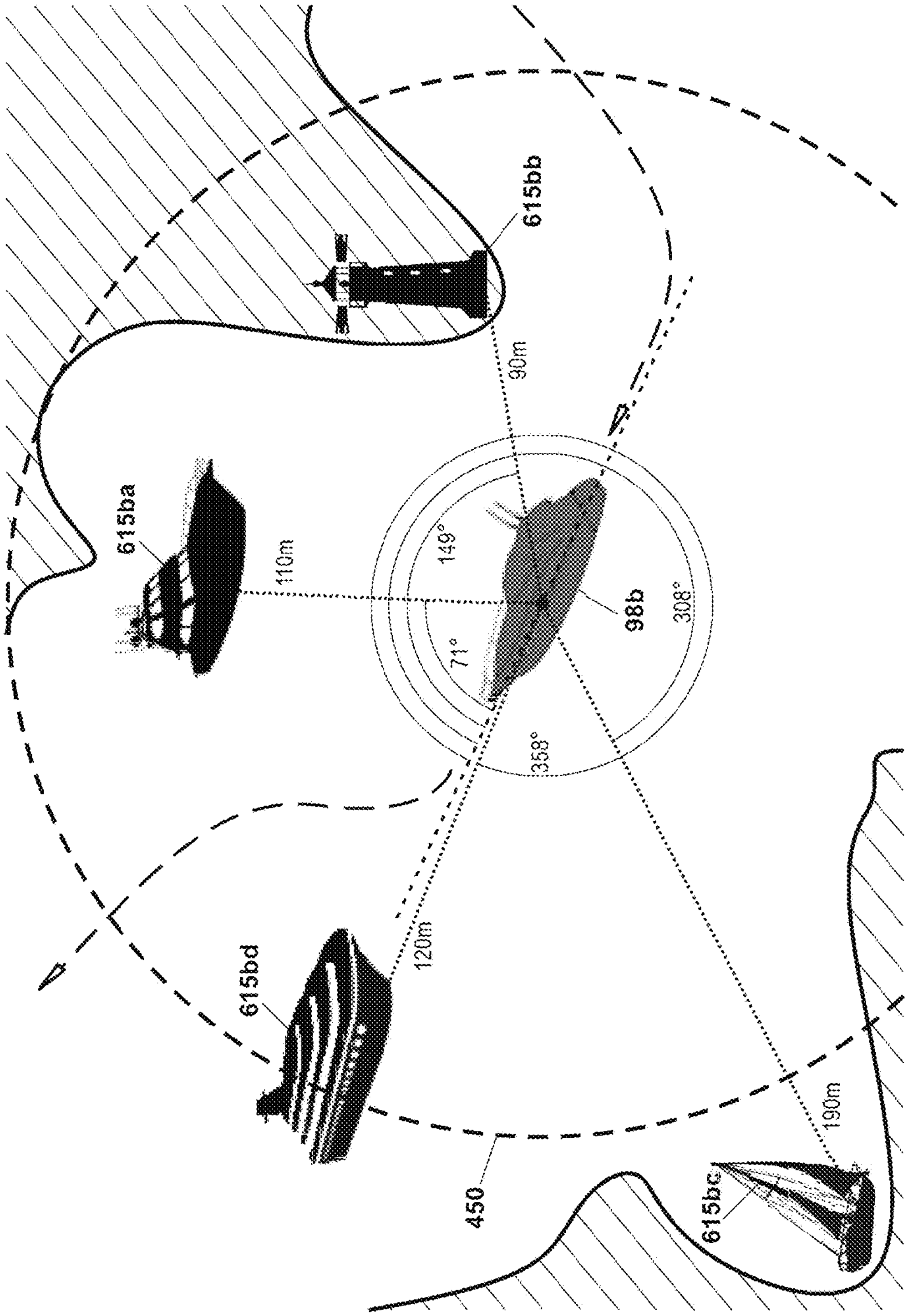

615bb
90m
615ba
110m
149°
98b
308°
71°
358°
120m
615bd
190m
450
615bc

FIG. 40

ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION

FIELD

The disclosure generally relates to computing enabled devices and/or systems. The disclosure includes devices, apparatuses, systems, and related methods for providing advanced learning, anticipating, decision making, automation, and/or other functionalities.

COPYRIGHT NOTICE

BACKGROUND

Devices or systems commonly operate by receiving a user's operating directions in various circumstances. Instructions are then executed to effect the operation of a device or system based on user's operating directions. Hence, devices or systems rely on the user to direct their behaviors. Commonly employed device or system operating techniques lack a way to learn operation of a device or system and enable autonomous operation of a device or system.

SUMMARY

In some aspects, the disclosure relates to a system for learning and using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises a processor circuit configured to execute instruction sets for operating a device. The system may further include a memory unit configured to store data. The system may further include a sensor configured to detect objects. The system may further include an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first collection of object representations, the first collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to learn the first collection of object representations correlated with the first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: receive a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations performed in response to the executing by the processor circuit.

In some embodiments, at least one of: the processor circuit, the memory unit, the sensor, or the artificial intelligence unit are part of, operating on, or coupled to the device. In further embodiments, the device includes one or more devices. In further embodiments, the device includes a construction machine, an assembly machine, an object handling machine, an object dispensing machine, a sorting machine, a restocking machine, an industrial machine, an agricultural machine, a harvesting machine, an appliance, a toy, a robot, a ground vehicle, an aerial vehicle, an aquatic vehicle, a computer, a smartphone, a control device, or a computing enabled device. In further embodiments, the processor circuit includes one or more processor circuits. In further embodiments, the processor circuit includes a logic circuit. The logic circuit may include a microcontroller. The one or more instruction sets may include one or more inputs into or one or more outputs from the logic circuit.

In certain embodiments, the processor circuit includes a logic circuit, the instruction sets for operating the device include inputs into the logic circuit, and executing instruction sets for operating the device includes performing logic operations on the inputs into the logic circuit and producing outputs for operating the device. The logic circuit includes a microcontroller.

In some embodiments, the processor circuit includes a logic circuit, the instruction sets for operating the device include outputs from the logic circuit for operating the device, and executing instruction sets for operating the device includes performing logic operations on inputs into the logic circuit and producing the outputs from the logic circuit for operating the device.

In certain embodiments, the memory unit includes one or more memory units. In further embodiments, the memory unit resides on a remote computing device or a remote computing system, the remote computing device or the remote computing system coupled to the processor circuit via a network or an interface. The remote computing device or the remote computing system may include a server, a cloud, a computing device, or a computing system accessible over the network or the interface.

In some embodiments, the sensor includes one or more sensors. In further embodiments, the sensor includes a camera, a microphone, a lidar, a radar, a sonar, or a detector. In further embodiments, the sensor is part of a remote device. In further embodiments, the sensor is configured to detect objects in the device's surrounding.

In certain embodiments, the artificial intelligence unit is coupled to the sensor. In further embodiments, the artificial intelligence unit is coupled to the memory unit. In further embodiments, the artificial intelligence unit is part of, operating on, or coupled to the processor circuit. In further embodiments, the system further comprises: an additional processor circuit, wherein the artificial intelligence unit is part of, operating on, or coupled to the additional processor circuit. In further embodiments, the artificial intelligence unit is part of, operating on, or coupled to a remote computing device or a remote computing system, the remote computing device or the remote computing system coupled

to the processor circuit via a network or an interface. In further embodiments, the artificial intelligence unit includes a circuit, a computing apparatus, or a computing system attachable to the processor circuit. In further embodiments, the artificial intelligence unit includes a circuit, a computing apparatus, or a computing system attachable to the device. In further embodiments, the artificial intelligence unit is attachable to an application for operating the device, the application running on the processor circuit. In further embodiments, the artificial intelligence unit includes a circuit, a computing apparatus, or a computing system built into the processor circuit. In further embodiments, the artificial intelligence unit includes a circuit, a computing apparatus, or a computing system built into the device. In further embodiments, the artificial intelligence unit is built into an application for operating the device, the application running on the processor circuit. In further embodiments, the artificial intelligence unit is provided as a feature of the processor circuit. In further embodiments, the artificial intelligence unit is provided as a feature of an application running on the processor circuit. In further embodiments, the artificial intelligence unit is provided as a feature of the device. In further embodiments, the artificial intelligence unit is further configured to: take control from, share control with, or release control to the processor circuit. In further embodiments, the artificial intelligence unit is further configured to: take control from, share control with, or release control to an application or an object of the application, the application running on the processor circuit.

In some embodiments, the first collection of object representations includes one or more representations of objects detected by the sensor at a time. In further embodiments, the new collection of object representations includes one or more representations of objects detected by the sensor at a time. In further embodiments, the first collection of object representations includes a stream of collections of object representations. In further embodiments, the new collection of object representations includes a stream of collections of object representations. In further embodiments, the first or the new collection of object representations includes or is associated with a time stamp, an order, or a time related information. In further embodiments, the first or the new collection of object representations includes one or more representations of objects in the device's surrounding. In further embodiments, the first or the new collection of object representations includes one or more representations of objects in a remote device's surrounding. In further embodiments, an object representation of the one or more object representations includes one or more object properties. In further embodiments, the first or the new collection of object representations includes one or more object properties. In further embodiments, the first collection of object representations includes a comparative collection of object representations whose at least one portion can be used for comparisons with at least one portion of collections of object representations subsequent to the first collection of object representations, the collections of object representations subsequent to the first collection of object representations comprising the new collection of object representations. In further embodiments, the first collection of object representations includes a comparative collection of object representations that can be used for comparisons with the new collection of object representations. In further embodiments, the new collection of object representations includes an anticipatory collection of object representations whose correlated one or more instruction sets can be used for anticipation of one or more instruction sets to be executed by the processor circuit.

In certain embodiments, the first one or more instruction sets for operating the device include one or more instruction sets that temporally correspond to the first collection of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first collection of object representations include one or more instruction sets executed at a time of generating the first collection of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first collection of object representations include one or more instruction sets executed prior to generating the first collection of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first collection of object representations include one or more instruction sets executed within a threshold period of time prior to generating the first collection of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first collection of object representations include one or more instruction sets executed subsequent to generating the first collection of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first collection of object representations include one or more instruction sets executed within a threshold period of time subsequent to generating the first collection of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first collection of object representations include one or more instruction sets executed within a threshold period of time prior to generating the first collection of object representations or a threshold period of time subsequent to generating the first collection of object representations.

In some embodiments, the first one or more instruction sets for operating the device are executed by the processor circuit. In further embodiments, the first one or more instruction sets for operating the device are part of an application for operating the device, the application running on the processor circuit. In further embodiments, the first one or more instruction sets for operating the device include one or more inputs into or one or more outputs from the processor circuit. In further embodiments, the first one or more instruction sets for operating the device include values or states of one or more registers or elements of the processor circuit. In further embodiments, the first one or more instruction sets for operating the device include at least one of: a command, a keyword, a symbol, an instruction, an operator, a variable, a value, an object, a data structure, a function, a parameter, a state, a signal, an input, an output, a character, a digit, or a reference thereto. In further embodiments, the first one or more instruction sets for operating the device include a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, a structured query language (SQL) code, or a machine code. In further embodiments, the first one or more instruction sets for operating the device include one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the processor circuit includes a logic circuit. The first one or more instruction sets for operating the device may include one or more inputs into a logic circuit. The first one or more instruction sets for operating the device may include one or more outputs from a logic circuit.

In certain embodiments, the first one or more instruction sets for operating the device include one or more instruction

sets for operating an application or an object of the application, the application running on the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes obtaining the first one or more instruction sets for operating the device from the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device as they are executed by the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from a register or an element of the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from an element that is part of, operating on, or coupled to the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from at least one of: the memory unit, the device, a virtual machine, a runtime engine, a hard drive, a storage device, a peripheral device, a network connected device, or a user. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from a plurality of processor circuits, applications, memory units, devices, virtual machines, runtime engines, hard drives, storage devices, peripheral devices, network connected devices, or users.

In some embodiments, the processor circuit includes a logic circuit, and wherein the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from the logic circuit. In further embodiments, the logic circuit includes a microcontroller. In further embodiments, the receiving the first one or more instruction sets for operating the device from the logic circuit includes receiving the first one or more instruction sets for operating the device from an element of the logic circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device from the logic circuit includes receiving one or more inputs into the logic circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device from the logic circuit includes receiving one or more outputs from the logic circuit.

In certain embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from an application for operating the device, the application running on the processor circuit.

In some embodiments, the system further comprises: an application including instruction sets for operating the device, the application running on the processor circuit, wherein the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from the application.

In certain embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, a structured query language (SQL) code, or a machine code. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of an element that is part of, operating on, or coupled to the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a register of the processor circuit, the memory unit, a storage, or a repository where the first one or more instruction sets for operating the device are stored. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of the processor circuit, the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a processing element. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of the processor circuit or tracing, profiling, or instrumentation of a component of the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of an application or an object of the application, the application running on the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of one or more of code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a user input. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: a manual, an automatic, a dynamic, or a just in time (JIT) tracing, profiling, or instrumentation. In further embodiments, the receiving the first one or more instruction sets for operating the device includes utilizing at least one of: a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, a logging tool, or an independent tool for obtaining instruction sets. In further embodiments, the receiving the first one or more instruction sets for operating the device includes utilizing an assembly language. In further embodiments, the receiving the first one or more instruction sets for operating the device includes utilizing a branch or a jump. In further embodiments, the receiving the first one or more instruction sets for operating the device includes a branch tracing or a simulation tracing.

In further embodiments, the system further comprises: an interface configured to receive instruction sets, wherein the first one or more instruction sets for operating the device are received by the interface. The interface may include an acquisition interface.

In some embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the device include a knowledge of how the device operated in a circumstance. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the device

are included in a neuron, a node, a vertex, or an element of a knowledgebase. In further embodiments, the knowledgebase includes a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. In further embodiments, some of the neurons, nodes, vertices, or elements are interconnected. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the device are structured into a knowledge cell. In further embodiments, the knowledge cell is included in a neuron, a node, a vertex, or an element of a knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device includes correlating the first collection of object representations with the first one or more instruction sets for operating the device. In further embodiments, the correlating the first collection of object representations with the first one or more instruction sets for operating the device includes generating a knowledge cell, the knowledge cell comprising the first collection of object representations correlated with the first one or more instruction sets for operating the device. In further embodiments, the correlating the first collection of object representations with the first one or more instruction sets for operating the device includes structuring a knowledge of how the device operated in a circumstance. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device includes learning a user's knowledge, style, or methodology of operating the device in a circumstance.

In certain embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device includes storing the first collection of object representations correlated with the first one or more instruction sets for operating the device into the memory unit, the first collection of object representations correlated with the first one or more instruction sets for operating the device being part of a plurality of collections of object representations correlated with one or more instruction sets for operating the device stored in the memory unit. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the device include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the device are organized into a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, each of the plurality of collections of object representations correlated with one or more instruction sets for operating the device is included in a neuron, a node, a vertex, or an element of a knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the device include a user's knowledge, style, or methodology of operating the device in circumstances. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the device are stored on a remote computing device or a remote computing system. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the device include an artificial intelligence system for knowledge structuring, storing, or representation. The artificial intelligence system for knowledge structuring, storing, or representation may include at least one of: a deep learning system, a supervised learning system, an unsupervised learning system, a neural network, a search-based system, an optimization-based system, a logic-based system, a fuzzy logic-based system, a tree-based system, a graph-based system, a hierarchical system, a symbolic system, a sub-symbolic system, an evolutionary system, a genetic system, a multi-agent system, a deterministic system, a probabilistic system, or a statistical system.

In some embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes comparing at least one portion of the new collection of object representations with at least one portion of the first collection of object representations. In further embodiments, the at least one portion of the new collection of object representations include at least one object representation or at least one object property of the new collection of object representations. In further embodiments, the at least one portion of the first collection of object representations include at least one object representation or at least one object property of the first collection of object representations. In further embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes comparing at least one object representation from the new collection of object representations with at least one object representation from the first collection of object representations. In further embodiments, the comparing at least one object representation from the new collection of object representations with at least one object representation from the first collection of object representations includes comparing at least one object property of the at least one object representation from the new collection of object representations with at least one object property of the at least one object representation from the first collection of object representations.

In certain embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes determining that there is at least a partial match between the new collection of object representations and the first collection of object representations. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that there is at least a partial match between one or more portions of the new collection of object representations and one or more portions of the first collection of object representations. In further embodiments, the determining that

there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that a similarity between at least one portion of the new collection of object representations and at least one portion of the first collection of object representations exceeds a similarity threshold. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining a substantial similarity between at least one portion of the new collection of object representations and at least one portion of the first collection of object representations. The substantial similarity may be achieved when a similarity between the at least one portion of the new collection of object representations and the at least one portion of the first collection of object representations exceeds a similarity threshold. The substantial similarity may be achieved when a number or a percentage of matching or partially matching portions of the new collection of object representations and portions of the first collection of object representations exceeds a threshold number or threshold percentage. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that a number or a percentage of matching or partially matching object representations from the new collection of object representations and from the first collection of object representations exceeds a threshold number or threshold percentage. The matching or partially matching object representations from the new collection of object representations and from the first collection of object representations may be determined factoring in at least one of a type of an object representation, an importance of an object representation, a threshold for a similarity in an object representation, or a threshold for a difference in an object representation. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that a number or a percentage of matching or partially matching object properties from the new collection of object representations and from the first collection of object representations exceeds a threshold number or threshold percentage. The matching or partially matching object properties from the new collection of object representations and from the first collection of object representations may be determined factoring in at least one of: an association of an object property with an object representation, a category of an object property, an importance of an object property, a threshold for a similarity in an object property, or a threshold for a difference in an object property. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that there is at least a partial match between at least one object representation from the new collection of object representations and at least one object representation from the first collection of object representations. The determining that there is at least a partial match between at least one object representation from the new collection of object representations and at least one object representation from the first collection of object representations includes determining that there is at least a partial match between at least one object property of the at least one object representation from the new collection of object representations and at least one object property of the at least one object representation from the first collection of object representations.

In certain embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations instead of or prior to an instruction set that would have been executed next. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first collection of object representations into a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting the processor circuit to the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting the processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes transmitting, to the processor circuit for execution, the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes issuing an interrupt to the processor circuit and executing the first one or more instruction sets for operating the device correlated with the first collection of object representations following the interrupt. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying an element that is part of, operating on, or coupled to the processor circuit.

In some embodiments, the processor circuit includes a logic circuit, and wherein the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the logic circuit includes a microcontroller. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying an element of the logic circuit. In further embodiments, the causing the logic

circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first collection of object representations into an element of the logic circuit. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting the logic circuit to the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes replacing inputs into the logic circuit with the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes replacing outputs from the logic circuit with the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In certain embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes causing an application for operating the device to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations, the application running on the processor circuit.

In some embodiments, the system further comprises: an application including instruction sets for operating the device, the application running on the processor circuit, wherein the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying the application.

In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting an application to the first one or more instruction sets for operating the device correlated with the first collection of object representations, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting an application to one or more alternate instruction sets, the application running on the processor circuit, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying at least one of: the memory unit, a register of the processor circuit, a storage, or a repository where instruction sets are stored or used. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets for operating an application or an object of the application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying at least one of: an element of the processor circuit, an element of the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing an assembly language. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the causing the processor circuit to

execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes adding or inserting additional code into a code of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes at least one of: modifying, removing, rewriting, or overwriting a code of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of an application, the application running on the processor circuit. The branching or redirecting the code may include inserting at least one of: a branch, a jump, or a means for redirecting an execution. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes implementing a user's knowledge, style, or methodology of operating the device in a circumstance.

In certain embodiments, the system further comprises: an interface configured to cause execution of instruction sets, wherein the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations is caused by the interface. The interface may include a modification interface.

In some embodiments, the one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations include one or more operations with or by a computing enabled device. In further embodiments, the performing the one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations includes implementing a user's knowledge, style, or methodology of operating the device in a circumstance.

In certain embodiments, the system further comprising: an application running on the processor circuit.

In some embodiments, the instruction sets for operating the device are part of an application for operating the device, the application running on the processor circuit.

In certain embodiments, the system of further comprises: an application for operating the device, the application running on the processor circuit. The application for operating the device may include the instruction sets for operating the device.

In some embodiments, the artificial intelligence unit is further configured to: receive at least one extra information. In further embodiments, the at least one extra information include one or more of: a time information, a location information, a computed information, or a contextual information. In further embodiments, the at least one extra information include one or more of: an information on an object, an information on an object representation, an information on a collection of object representations, an information on a device's circumstance, an information on an instruction set, an information on an application, an information on the processor circuit, an information on the device, or an information on an user. In further embodiments, the artificial intelligence unit is further configured to: learn the first collection of object representations correlated with the at least one extra information. The learning the first collection of object representations correlated with at least one extra information may include correlating the first collection of object representations with the at least one extra information. The learning the first collection of object representations correlated with at least one extra information may include storing the first collection of object representations correlated with the at least one extra information into the memory unit. In further embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations. The anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations may include comparing an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations. The anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations may include determining that a similarity between an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations exceeds a similarity threshold.

In certain embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: cause the user interface to present a user with an option to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In some embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: receive, via the user interface, a user's selection to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In certain embodiments, the artificial intelligence unit is further configured to: rate the executed first one or more instruction sets for operating the device correlated with the first collection of object representations. The rating the executed first one or more instruction sets for operating the device correlated with the first collection of object representations may include causing a user interface to display the executed first one or more instruction sets for operating the device correlated with the first collection of object representations along with one or more rating values as options to be selected by a user. The rating the executed first one or more instruction sets for operating the device correlated with the first collection of object representations may

include rating the executed first one or more instruction sets for operating the device correlated with the first collection of object representations without a user input.

In some embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: cause the user interface to present a user with an option to cancel the execution of the executed first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the canceling the execution of the executed first one or more instruction sets for operating the device correlated with the first collection of object representations includes restoring the processor circuit or the device to a prior state. The restoring the processor circuit or the device to a prior state may include saving the state of the processor circuit or the device prior to executing the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In certain embodiments, the system further comprises: an input device configured to receive a user's operating directions, the user's operating directions for instructing the processor circuit on how to operate the device.

In some embodiments, the autonomous device operating includes a partially or a fully autonomous device operating. The partially autonomous device operating may include executing the first one or more instruction sets for operating the device correlated with the first collection of object representations responsive to a user confirmation. In further embodiments, the fully autonomous device operating may include executing the first one or more instruction sets for operating the device correlated with the first collection of object representations without a user confirmation.

In certain embodiments, the artificial intelligence unit is further configured to: receive a second collection of object representations, the second collection of object representations including one or more representations of objects detected by the sensor; receive a second one or more instruction sets for operating the device; and learn the second collection of object representations correlated with the second one or more instruction sets for operating the device. In further embodiments, the second collection of object representations includes one or more representations of objects detected by the sensor at a time. In further embodiments, the second collection of object representations includes a stream of collections of object representations. In further embodiments, the second collection of object representations includes or is associated with a time stamp, an order, or a time related information. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device and the learning the second collection of object representations correlated with the second one or more instruction sets for operating the device include creating a connection between the first collection of object representations correlated with the first one or more instruction sets for operating the device and the second collection of object representations correlated with the second one or more instruction sets for operating the device. The connection may include or be associated with at least one of: an occurrence count, a weight, a parameter, or a data. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device and the learning the second collection of object representations correlated with the second one or more instruction sets for operating the device include updating a connection between the first collection of object representations correlated with the first one or more instruction sets for operating the device and the second collection of object representations correlated with the second one or more instruction sets for operating the device. The updating the connection between the first collection of object representations correlated with the first one or more instruction sets for operating the device and the second collection of object representations correlated with the second one or more instruction sets for operating the device may include updating at least one of: an occurrence count, a weight, a parameter, or a data included in or associated with the connection. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device includes storing the first collection of object representations correlated with the first one or more instruction sets for operating the device into a first node of a knowledgebase, and wherein the learning the second collection of object representations correlated with the second one or more instruction sets for operating the device includes storing the second collection of object representations correlated with the second one or more instruction sets for operating the device into a second node of the knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. The knowledgebase may be stored in the memory unit. The learning the first collection of object representations correlated with the first one or more instruction sets for operating the device and the learning the second collection of object representations correlated with the second one or more instruction sets for operating the device include creating a connection between the first node and the second node. The learning the first collection of object representations correlated with the first one or more instruction sets for operating the device and the learning the second collection of object representations correlated with the second one or more instruction sets for operating the device include updating a connection between the first node and the second node. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the device is stored into a first node of a neural network and the second collection of object representations correlated with the second one or more instruction sets for operating the device is stored into a second node of the neural network. The first node and the second node may be connected by a connection. The first node may be part of a first layer of the neural network and the second node may be part of a second layer of the neural network. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the device is stored into a first node of a graph and the second collection of object representations correlated with the second one or more instruction sets for operating the device is stored into a second node of the graph. The first node and the second node may be connected by a connection. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the device is stored into a first node of a sequence and the second collection of object representations correlated with the second one or more instruction sets for operating the device is stored into a second node of the sequence.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising:

receiving a first collection of object representations, the first collection of object representations including one or more representations of objects detected by a sensor. The operations may further include: receiving a first one or more instruction sets for operating a device. The operations may further include: learning the first collection of object representations correlated with the first one or more instruction sets for operating the device. The operations may further include: receiving a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The operations may further include: anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further include: causing an execution of the first one or more instruction sets for operating the device correlated with the first collection of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations by the device performed in response to the execution.

In certain embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from the one or more processor circuits or from another one or more processor circuits. In further embodiments, the execution of the first one or more instruction sets for operating the device correlated with the first collection of object representations is performed by the one or more processor circuits or by another one or more processor circuits.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a processor circuit, the first collection of object representations including one or more representations of objects detected by a sensor. The method may further include: (b) receiving a first one or more instruction sets for operating a device by the processor circuit. The method may further include: (c) learning the first collection of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the processor circuit. The method may further include: (d) receiving a new collection of object representations by the processor circuit, the new collection of object representations including one or more representations of objects detected by the sensor. The method may further include: (e) anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (e) performed by the processor circuit. The method may further include: (f) executing the first one or more instruction sets for operating the device correlated with the first collection of object representations, the executing of (f) performed in response to the anticipating of (e). The method may further include: (g) performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations by the device performed in response to the executing of (f).

In certain embodiments, the receiving of (b) includes receiving the first one or more instruction sets for operating the device from the processor circuit or from another processor circuit. In further embodiments, the executing of (f) is performed by the processor circuit or by another processor circuit.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable as well as the following embodiments.

In some embodiments, the first one or more instruction sets for operating the device are executed by a processor circuit. In further embodiments, the first one or more instruction sets for operating the device are part of an application for operating the device. In further embodiments, the first one or more instruction sets for operating the device include one or more inputs into or one or more outputs from a processor circuit. In further embodiments, the first one or more instruction sets for operating the device include values or states of one or more registers or elements of a processor circuit. In further embodiments, the first one or more instruction sets for operating the device include one or more inputs into a logic circuit. In further embodiments, the first one or more instruction sets for operating the device include one or more outputs from a logic circuit. In further embodiments, the first one or more instruction sets for operating the device include one or more instruction sets for operating an application or an object of the application.

In certain embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device as they are executed by a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from a register or an element of a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from an element that is part of, operating on, or coupled to a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from at least one of: the memory unit, the device, a virtual machine, a runtime engine, a hard drive, a storage device, a peripheral device, a network connected device, or a user. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from a plurality of processor circuits, applications, memory units, devices, virtual machines, runtime engines, hard drives, storage devices, peripheral devices, network connected devices, or users.

In some embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from a logic circuit. In further embodiments, the logic circuit

includes a microcontroller. In further embodiments, the receiving the first one or more instruction sets for operating the device from the logic circuit includes receiving the first one or more instruction sets for operating the device from an element of the logic circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device from the logic circuit includes receiving one or more inputs into the logic circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device from the logic circuit includes receiving one or more outputs from the logic circuit.

In certain embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from an application for operating the device. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from an application, the application including instruction sets for operating the device. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of an element that is part of, operating on, or coupled to a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a register of a processor circuit, the memory unit, a storage, or a repository where the first one or more instruction sets for operating the device are stored. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a processor circuit, the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a processing element. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a processor circuit or tracing, profiling, or instrumentation of a component of a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of an application or an object of the application. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device by an interface. The interface may include an acquisition interface.

In some embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device includes storing the first collection of object representations correlated with the first one or more instruction sets for operating the device into a memory unit, the first collection of object representations correlated with the first one or more instruction sets for operating the device being part of a plurality of collections of object representations correlated with one or more instruction sets for operating the device stored in the memory unit.

In certain embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes executing the first one or more instruction sets for operating the device correlated with the first collection of object representations instead of or prior to an instruction set that would have been executed next. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying a register or an element of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first collection of object representations into a register or an element of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting a processor circuit to the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting a processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes transmitting, to a processor circuit for execution, the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes issuing an interrupt to a processor circuit and executing the first one or more instruction sets for operating the device correlated with the first collection of object representations following the interrupt. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying an element that is part of, operating on, or coupled to a processor circuit.

In some embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes executing, by a logic circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the logic circuit includes a microcontroller. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying an element of the logic circuit. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first collection of object representations into an element of the logic circuit. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting the logic circuit to the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations includes replacing inputs into the logic circuit with the first one or more instruction sets for

operating the device correlated with the first collection of object representations. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations includes replacing outputs from the logic circuit with the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In certain embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes executing, by an application for operating the device, the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying an application, the application including instruction sets for operating the device. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting an application to the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting an application to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying at least one of: the memory unit, a register of a processor circuit, a storage, or a repository where instruction sets are stored or used. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets for operating an application or an object of the application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying at least one of: an element of a processor circuit, an element of the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing an assembly language. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes adding or inserting additional code into a code of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes at least one of: modifying, removing, rewriting, or overwriting a code of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of an application. The branching or redirecting the code may include inserting at least one of: a branch, a jump, or a means for redirecting an execution. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes implementing a user's knowledge, style, or methodology of operating the device in a circumstance. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes executing the first one or more instruction sets for operating the device correlated with the first collection of object representations via an interface. The interface may include a modification interface.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving at least one extra information. In further

embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: learning the first collection of object representations correlated with the at least one extra information.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: presenting, via a user interface, a user with an option to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving, via a user interface, a user's selection to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: rating the executed first one or more instruction sets for operating the device correlated with the first collection of object representations.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: presenting, via a user interface, a user with an option to cancel the execution of the executed first one or more instruction sets for operating the device correlated with the first collection of object representations.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving, via an input device, a user's operating directions, the user's operating directions for instructing a processor circuit on how to operate the device.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving a second collection of object representations, the second collection of object representations including one or more representations of objects detected by the sensor receiving a second one or more instruction sets for operating the device; and learning the second collection of object representations correlated with the second one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving, by a first processor circuit of the one or more processor circuits, a first collection of object representations, the first collection of object representations including one or more representations of objects detected by a sensor. The operations may further comprise: receiving, by the first processor circuit of the one or more processor circuits, a first one or more instruction sets for operating a device. The operations may further comprise: learning, by the first processor circuit of the one or more processor circuits, the first collection of object representations correlated with the first one or more instruction sets for operating the device. The operations may further comprise: receiving, by the first processor circuit of the one or more processor circuits, a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The operations may further comprise: anticipating, by the first processor circuit of the one or more processor circuits, the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further comprise: causing, by the first processor circuit of the one or more processor circuits, an execution, by a second processor circuit of the one or more processor circuits, of the first one or more instruction sets for operating the device correlated with the first collection of object representations, the causing performed in response to the anticipating, by the first processor circuit of the one or more processor circuits, the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations by the device performed in response to the execution.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a first processor circuit, the first collection of object representations including one or more representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more instruction sets for operating a device by the first processor circuit. The method may further comprise: (c) learning the first collection of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the first processor circuit. The method may further comprise: (d) receiving a new collection of object representations by the first processor circuit, the new collection of object representations including one or more representations of objects detected by the sensor. The method may further comprise: (e) anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (e) performed by the first processor circuit. The method may further comprise: (f) executing, by a second processor circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations, the executing of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations by the device performed in response to the executing of (f).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for learning a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit

may be configured to: receive a first collection of object representations, the first collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: learn the first collection of object representations correlated with the first one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first collection of object representations, the first collection of object representations including one or more representations of objects detected by a sensor. The operations may further comprise: receiving a first one or more instruction sets for operating a device. The operations may further comprise: learning the first collection of object representations correlated with the first one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a processor circuit, the first collection of object representations including one or more representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more instruction sets for operating a device by the processor circuit. The method may further comprise: (c) learning the first collection of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the processor circuit.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: access the memory unit that comprises a plurality of collections of object representations correlated with one or more instruction sets for operating the device, the plurality of collections of object representations correlated with one or more instruction sets for operating the device including a first collection of object representations correlated with a first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: receive a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations performed in response to the executing by the processor circuit.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: accessing a memory unit that comprises a plurality of collections of object representations correlated with one or more instruction sets for operating a device, the plurality of collections of object representations correlated with one or more instruction sets for operating the device including a first collection of object representations correlated with a first one or more instruction sets for operating the device. The operations may further comprise: receiving a new collection of object representations, the new collection of object representations including one or more representations of objects detected by a sensor. The operations may further comprise: anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further comprise: causing an execution of the first one or more instruction sets for operating the device correlated with the first collection of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations by the device performed in response to the execution.

In some aspects, the disclosure relates to a method comprising: (a) accessing a memory unit that comprises a plurality of collections of object representations correlated with one or more instruction sets for operating a device, the plurality of collections of object representations correlated with one or more instruction sets for operating the device including a first collection of object representations correlated with a first one or more instruction sets for operating the device, the accessing of (a) performed by a processor circuit. The method may further comprise: (b) receiving a new collection of object representations by the processor circuit, the new collection of object representations including one or more representations of objects detected by a sensor. The method may further comprise: (c) anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (c) performed by the processor circuit. The method may further comprise: (d) executing the first one or more instruction sets for operating the device correlated with the first collection of object

representations, the executing of (d) performed in response to the anticipating of (c). The method may further comprise: (e) performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations by the device performed in response to the executing of (d).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for learning and using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first stream of collections of object representations, the first stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: learn the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: receive a new stream of collections of object representations, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations performed in response to the executing by the processor circuit.

In certain embodiments, each collection of object representations includes one or more representations of objects detected by the sensor at a time. In further embodiments, each collection of object representations includes one or more of object representations. In further embodiments, each collection of object representations includes or is associated with a time stamp, an order, or a time related information. In further embodiments, the first stream of collections of object representations includes one or more collections of representations of objects detected by the sensor over time. In further embodiments, the new stream of collections of object representations includes one or more collections of representations of objects detected by the sensor over time. In further embodiments, the first or the new stream of collections of object representations includes one or more collections of representations of objects in the device's surrounding. In further embodiments, the first or the new stream of collections of object representations includes one or more collections of representations of objects in a remote device's surrounding. In further embodiments, an object representation of a stream of collections of object representations includes one or more object properties. In further embodiments, the first or the new stream of collections of object representations includes one or more object properties. In further embodiments, the first stream of collections of object representations includes a comparative stream of collections of object representations whose at least one portion can be used for comparisons with at least one portion of streams of collections of object representations subsequent to the first stream of collections of object representations, the streams of collections of object representations subsequent to the first stream of collections of object representations comprising the new stream of collections of object representations. In further embodiments, the first stream of collections of object representations includes a comparative stream of collections of object representations that can be used for comparisons with the new stream of collections of object representations. In further embodiments, the new stream of collections of object representations includes an anticipatory stream of collections of object representations whose correlated one or more instruction sets can be used for anticipation of one or more instruction sets to be executed by the processor circuit.

In some embodiments, the first one or more instruction sets for operating the device include one or more instruction sets that temporally correspond to the first stream of collections of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first stream of collections of object representations include one or more instruction sets executed at a time of generating the first stream of collections of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first stream of collections of object representations include one or more instruction sets executed prior to generating the first stream of collections of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first stream of collections of object representations include one or more instruction sets executed within a threshold period of time prior to generating the first stream of collections of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first stream of collections of object representations include one or more instruction sets executed subsequent to generating the first stream of collections of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first stream of collections of object representations include one or more instruction sets executed within a threshold period of time subsequent to generating the first stream of collections of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first stream of collections of object representations include one or more instruction sets executed within a threshold period of time prior to generating the first stream of collections of object representations or a threshold period of time subsequent to generating the first stream of collections of object representations.

In certain embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device include a knowledge of how the device operated in a circumstance. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device are included in a neuron, a node, a vertex, or an element of a knowledgebase. In further embodiments, the knowledgebase includes a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. In further embodiments, some of the neurons, nodes, vertices, or elements are interconnected. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device are structured into a knowledge cell. In further embodiments, the knowledge cell is included in a neuron, a node, a vertex, or an element of a knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device includes correlating the first stream of collections of object representations with the first one or more instruction sets for operating the device. In further embodiments, the correlating the first stream of collections of object representations with the first one or more instruction sets for operating the device includes generating a knowledge cell, the knowledge cell comprising the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device. In further embodiments, the correlating the first stream of collections of object representations with the first one or more instruction sets for operating the device includes structuring a knowledge of how the device operated in a circumstance. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device includes learning a user's knowledge, style, or methodology of operating the device in a circumstance.

In some embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device includes storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device into the memory unit, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device being part of a plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device stored in the memory unit. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device are organized into a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, each stream of collections of object representations correlated with one or more instruction sets for operating the device of the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device is included in a neuron, a node, a vertex, or an element of a knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device include a user's knowledge, style, or methodology of operating the device in circumstances. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device are stored on a remote computing device or a remote computing system. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device include an artificial intelligence system for knowledge structuring, storing, or representation. The artificial intelligence system for knowledge structuring, storing, or representation may include at least one of: a deep learning system, a supervised learning system, an unsupervised learning system, a neural network, a search-based system, an optimization-based system, a logic-based system, a fuzzy logic-based system, a tree-based system, a graph-based system, a hierarchical system, a symbolic system, a sub-symbolic system, an evolutionary system, a genetic system, a multi-agent system, a deterministic system, a probabilistic system, or a statistical system.

In certain embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes comparing at least one portion of the new stream of collections of object representations with at least one portion of the first stream of collections of object representations. In further embodiments, the at least one portion of the new stream of collections of object representations include at least one collection of object representations, at least one object representation, or at least one object property of the new stream of collections of object representations. In further embodiments, the at least one portion of the first stream of collections of object representations include at least one collection of object representations, at least one object representation, or at least one object property of the first stream of collections of object representations. In further embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes comparing at least one collection of object representations from the new stream of collections of object representations with at least one collection of object representations from the first stream of collections of object representations. In further embodiments, the comparing at least one collection of object representations from the new stream of collections of object representations with at least one collection of object representations from the first stream of collections of object representations includes comparing at least one object representation of the at least one collection of object representations from the new stream of collections of object

representations with at least one object representation of the at least one collection of object representations from the first stream of collections of object representations. The comparing at least one object representation of the at least one collection of object representations from the new stream of collections of object representations with at least one object representation of the at least one collection of object representations from the first stream of collections of object representations may include comparing at least one object property of the at least one object representation of the at least one collection of object representations from the new stream of collections of object representations with at least one object property of the at least one object representation of the at least one collection of object representations from the first stream of collections of object representations.

In some embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that there is at least a partial match between one or more portions of the new stream of collections of object representations and one or more portions of the first stream of collections of object representations. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that a similarity between at least one portion of the new stream of collections of object representations and at least one portion of the first stream of collections of object representations exceeds a similarity threshold. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining a substantial similarity between at least one portion of the new stream of collections of object representations and at least one portion of the first stream of collections of object representations. The substantial similarity may be achieved when a similarity between the at least one portion of the new stream of collections of object representations and the at least one portion of the first stream of collections of object representations exceeds a similarity threshold. The substantial similarity may be achieved when a number or a percentage of matching or partially matching portions of the new stream of collections of object representations and portions of the first stream of collections of object representations exceeds a threshold number or threshold percentage. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that a number or a percentage of matching or partially matching collections of object representations from the new stream of collections of object representations and from the first stream of collections of object representations exceeds a threshold number or threshold percentage. The matching or partially matching collections of object representations from the new stream of collections of object representations and from the first stream of collections of

object representations may be determined factoring in at least one of: an importance of a collection of object representations, an order of a collection of object representations, a threshold for a similarity in a collection of object representations, or a threshold for a difference in a collection of object representations. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that a number or a percentage of matching or partially matching object representations from the new stream of collections of object representations and from the first stream of collections of object representations exceeds a threshold number or threshold percentage. The matching or partially matching object representations from the new stream of collections of object representations and from the first stream of collections of object representations may be determined factoring in at least one of: an association of an object representation with a collection of object representations, a type of an object representation, an importance of an object representation, a threshold for a similarity in an object representation, or a threshold for a difference in an object representation. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that a number or a percentage of matching or partially matching object properties from the new stream of collections of object representations and from the first stream of collections of object representations exceeds a threshold number or threshold percentage. The matching or partially matching object properties from the new stream of collections of object representations and from the first stream of collections of object representations may be determined factoring in at least one of: an association of an object property with an object representation, an association of an object property with a collection of object representations, a category of an object property, an importance of an object property, a threshold for a similarity in an object property, or a threshold for a difference in an object property. In further embodiments, determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that there is at least a partial match between at least one collection of object representations from the new stream of collections of object representations and at least one collection of object representations from the first stream of collections of object representations. The determining that there is at least a partial match between at least one collection of object representations from the new stream of collections of object representations and at least one collection of object representations from the first stream of collections of object representations may include determining that there is at least a partial match between at least one object representation of the at least one collection of object representations from the new stream of collections of object representations and at least one object representation of the at least one collection of object representations from the first stream of collections of object representations. The determining that there is at least a partial match between at least one object representation of the at least one collection of object representations from the new stream of collections of object representations and at least one object representation of the at least one collection of object representations from the first stream of collections of object representations may include determining that there is at least a partial match between at least one

object property of the at least one object representation of the at least one collection of object representations from the new stream of collections of object representations and at least one object property of the at least one object representation of the at least one collection of object representations from the first stream of collections of object representations.

In some embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations instead of or prior to an instruction set that would have been executed next. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations into a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting the processor circuit to the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting the processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes transmitting, to the processor circuit for execution, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes issuing an interrupt to the processor circuit and executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations following the interrupt. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying an element that is part of, operating on, or coupled to the processor circuit.

In certain embodiments, the processor circuit includes a logic circuit, and wherein the causing the processor circuit to execute the first one or more instruction sets for operating

the device correlated with the first stream of collections of object representations includes causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the logic circuit includes a microcontroller. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying an element of the logic circuit. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations into an element of the logic circuit. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting the logic circuit to the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes replacing inputs into the logic circuit with the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes replacing outputs from the logic circuit with the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In some embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes causing an application for operating the device to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the application running on the processor circuit.

In certain embodiments, the system further comprises: an application including instruction sets for operating the device, the application running on the processor circuit, wherein the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying the application.

In some embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting an application to the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting an application to one or more alternate instruction sets, the application running on the processor circuit, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodi-

ments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying at least one of: the memory unit, a register of the processor circuit, a storage, or a repository where instruction sets are stored or used. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets for operating an application or an object of the application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying at least one of: an element of the processor circuit, an element of the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing an assembly language. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes adding or inserting additional code into a code of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes at least one of: modifying, removing, rewriting, or overwriting a code of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of an application, the application running on the processor circuit. The branching or redirecting the code may include inserting at least one of: a branch, a jump, or a means for redirecting an execution. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes implementing a user's knowledge, style, or methodology of operating the device in a circumstance.

In certain embodiments, the system further comprises: an interface configured to cause execution of instruction sets, wherein the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations is caused by the interface. The interface may include a modification interface.

In some embodiments, the performing the one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes implementing a user's knowledge, style, or methodology of operating the device in a circumstance.

In certain embodiments, the system further comprises: an application running on the processor circuit.

In some embodiments, the instruction sets for operating the device are part of an application for operating the device, the application running on the processor circuit.

In certain embodiments, the system further comprises: an application for operating the device, the application running on the processor circuit. The application for operating the device may include the instruction sets for operating the device.

In some embodiments, the artificial intelligence unit is further configured to: receive at least one extra information. In further embodiments, the at least one extra information include one or more of: a time information, a location information, a computed information, or a contextual information. In further embodiments, the at least one extra information include one or more of: an information on an object, an information on an object representation, an information on a collection of object representations, an information on a stream of collections of object representations, an information on a device's circumstance, an information on an instruction set, an information on an application, an information on the processor circuit, an information on the device, or an information on an user. In further embodiments, the artificial intelligence unit is further configured to: learn the first stream of collections of object representations correlated with the at least one extra information. The learning the first stream of collections of object representations correlated with at least one extra information may include correlating the first stream of collections of object representations with the at least one extra information. The learning the first stream of collections of object representations correlated with at least one extra information may include storing the first stream of collections of object representations correlated with the at least one extra information into the memory unit. In further embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations. The anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations may include comparing an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations. The anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations may include determining that a similarity between an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations exceeds a similarity threshold.

In certain embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: cause the user interface to present a user with an option to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In some embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: receive, via the user interface, a user's selection to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In certain embodiments, the artificial intelligence unit is further configured to: rate the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. The rating the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations may include causing a user interface to display the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations along with one or more rating values as options to be selected by a user. The rating the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations may include rating the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations without a user input.

In some embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: cause the user interface to present a user with an option to cancel the execution of the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the canceling the execution of the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes restoring the processor circuit or the device to a prior state. The restoring the processor circuit or the device to a prior state may include saving the state of the processor circuit or the device prior to executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In certain embodiments, the system further comprises: an input device configured to receive a user's operating directions, the user's operating directions for instructing the processor circuit on how to operate the device.

In some embodiments, the autonomous device operating includes a partially or a fully autonomous device operating. The partially autonomous device operating may include executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations responsive to a user confirmation. The fully autonomous device operating may include executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations without a user confirmation.

In certain embodiments, the artificial intelligence unit is further configured to: receive a second stream of collections of object representations, the second stream of collections of object representations including one or more collections of representations of objects detected by the sensor; receive a second one or more instruction sets for operating the device; and learn the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device. In further embodiments, the second stream of collections of object representations includes one or more collections of representations of objects detected by the sensor over time. In further embodiments, the second stream of collections of object representations includes or is associated with a time stamp, an order,

or a time related information. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device include creating a connection between the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device. The connection may include or is associated with at least one of: an occurrence count, a weight, a parameter, or a data. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device include updating a connection between the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device. In further embodiments, the updating the connection between the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device includes updating at least one of: an occurrence count, a weight, a parameter, or a data included in or associated with the connection. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device includes storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device into a first node of a knowledgebase, and wherein the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device includes storing the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device into a second node of the knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. In further embodiments, the knowledgebase may be stored in the memory unit. The learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device may include creating a connection between the first node and the second node. The learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device may include updating a connection between the first node and the second node. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device is stored into a first node of a neural network and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device is stored into a second node of the neural network. The first node and the second node may be connected by a connection. The first node may be part of a first layer of the neural network and the second node may be part of a second layer of the neural network. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device is stored into a first node of a graph and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device is stored into a second node of the graph. The first node and the second node may be connected by a connection. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device is stored into a first node of a sequence and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device is stored into a second node of the sequence.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first stream of collections of object representations, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The operations may further comprise: receiving a first one or more instruction sets for operating a device. The operations may further comprise: learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device. The operations may further comprise: receiving a new stream of collections of object representations, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The operations may further comprise: anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The operations may further comprise: causing an execution of the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the execution.

In certain embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from the one or more processor circuits or from another one or more processor circuits. In further embodiments, the execution of the first one or more instruction sets for operating the device correlated with the first stream of

collections of object representations is performed by the one or more processor circuits or by another one or more processor circuits.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first stream of collections of object representations by a processor circuit, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more instruction sets for operating a device by the processor circuit. The method may further comprise: (c) learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the processor circuit. The method may further comprise: (d) receiving a new stream of collections of object representations by the processor circuit, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The method may further comprise: (e) anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, the anticipating of (e) performed by the processor circuit. The method may further comprise: (f) executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the executing of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the executing of (f).

In some embodiments, the receiving of (b) includes receiving the first one or more instruction sets for operating the device from the processor circuit or from another processor circuit. In further embodiments, the executing of (f) is performed by the processor circuit or by another processor circuit.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable as well as the following embodiments.

In certain embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device includes storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device into a memory unit, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device being part of a plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device stored in the memory unit.

In some embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations instead of or prior to an instruction set

that would have been executed next. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying a register or an element of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations into a register or an element of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting a processor circuit to the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting a processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes transmitting, to a processor circuit for execution, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes issuing an interrupt to a processor circuit and executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations following the interrupt. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying an element that is part of, operating on, or coupled to a processor circuit.

In certain embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes executing, by a logic circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the logic circuit includes a microcontroller. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying an element of the logic circuit. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations into an element of the logic circuit. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object represen-

tations includes redirecting the logic circuit to the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes replacing inputs into the logic circuit with the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes replacing outputs from the logic circuit with the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In some embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes executing, by an application for operating the device, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying an application, the application including instruction sets for operating the device. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting an application to the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting an application to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying at least one of: the memory unit, a register of a processor circuit, a storage, or a repository where instruction sets are stored or used. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets for operating an application or an object of the application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying at least one of: an element of a processor circuit, an element of the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing an assembly language. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes adding or inserting additional code into a code of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes at least one of: modifying, removing, rewriting, or overwriting a code of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of an application. The branching or redirecting the code may include inserting at least one of: a branch, a jump, or a means for redirecting an execution. In further embodiments, the

executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes implementing a user's knowledge, style, or methodology of operating the device in a circumstance. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations via an interface. The interface may include a modification interface.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving at least one extra information. In further embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: learning the first stream of collections of object representations correlated with the at least one extra information.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: presenting, via a user interface, a user with an option to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving, via a user interface, a user's selection to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: rating the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: presenting, via a user interface, a user with an option to cancel the execution of the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving, via an input device, a user's operating directions, the user's operating directions for instructing a processor circuit on how to operate the device.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving a second stream of collections of object representations, the second stream of collections of object representations including one or more collections of representations of objects detected by the sensor; receiving a second one or more instruction sets for operating the device; and learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving, by a first processor circuit of the one or more processor circuits, a first stream of collections of object representations, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The operations may further comprise: receiving, by the first processor circuit of the one or more processor circuits, a first one or more instruction sets for operating a device. The operations may further comprise: learning, by the first processor circuit of the one or more processor circuits, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device. The operations may further comprise: receiving, by the first processor circuit of the one or more processor circuits, a new stream of collections of object representations, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The operations may further comprise: anticipating, by the first processor circuit of the one or more processor circuits, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The operations may further comprise: causing, by the first processor circuit of the one or more processor circuits, an execution, by a second processor circuit of the one or more processor circuits, of the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the causing performed in response to the anticipating, by the first processor circuit of the one or more processor circuits, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the execution.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first stream of collections of object representations by a first processor circuit, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more instruction sets for operating a device by the first processor circuit. The method may further comprise: (c) learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the first processor circuit. The method may further comprise: (d) receiving a new stream of collections of object representations by the first processor circuit, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The method may further comprise: (e) anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, the anticipating of (e) performed by the first processor circuit. The method may further comprise: (f) executing, by a second processor circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the executing of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the device, one or more operations defined by the first one or more instruction

sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the executing of (f).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for learning a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first stream of collections of object representations, the first stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: learn the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first stream of collections of object representations, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The operations may further comprise: receiving a first one or more instruction sets for operating a device. The operations may further comprise: learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a non method comprising: (a) receiving a first stream of collections of object representations by a processor circuit, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more instruction sets for operating a device by the processor circuit. The method may further comprise: (c) learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the processor circuit.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: access the memory unit that comprises a plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device including a first stream of collections of object representations correlated with a first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: receive a new stream of collections of object representations, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations performed in response to the executing by the processor circuit.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: accessing a memory unit that comprises a plurality of streams of collections of object representations correlated with one or more instruction sets for operating a device, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device including a first stream of collections of object representations correlated with a first one or more instruction sets for operating the device. The operations may further comprise: receiving a new stream of collections of object representations, the new stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The operations may further comprise: anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The operations may further comprise: causing an execution of the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the device corre-

lated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the execution.

In some aspects, the disclosure relates to a method comprising: (a) accessing a memory unit that comprises a plurality of streams of collections of object representations correlated with one or more instruction sets for operating a device, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device including a first stream of collections of object representations correlated with a first one or more instruction sets for operating the device, the accessing of (a) performed by a processor circuit. The method may further comprise: (b) receiving a new stream of collections of object representations by the processor circuit, the new stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The method may further comprise: (c) anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, the anticipating of (c) performed by the processor circuit. The method may further comprise: (d) executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the executing of (d) performed in response to the anticipating of (c). The method may further comprise: (e) performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the executing of (d).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for learning and using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a logic circuit configured to receive inputs and produce outputs, wherein the outputs are used for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first collection of object representations, the first collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more inputs, wherein the first one or more inputs are also received by the logic circuit. The artificial intelligence unit may be further configured to: learn the first collection of object representations correlated with the first one or more inputs. The artificial intelligence unit may be further configured to: receive a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may be further configured to: cause the logic circuit to receive the first one or more inputs correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by one or more outputs for operating the device produced by the logic circuit.

In some embodiments, the logic circuit configured to receive inputs and produce outputs includes a logic circuit configured to produce outputs based at least in part on logic operations performed on the inputs. In further embodiments, the learning the first collection of object representations correlated with the first one or more inputs includes correlating the first collection of object representations with the first one or more inputs. In further embodiments, the learning the first collection of object representations correlated with the first one or more inputs includes storing the first collection of object representations correlated with the first one or more inputs into the memory unit, the first collection of object representations correlated with the first one or more inputs being part of a plurality of collections of object representations correlated with one or more inputs stored in the memory unit. In further embodiments, the anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes comparing at least one portion of the new collection of object representations with at least one portion of the first collection of object representations. In further embodiments, the anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes determining that there is at least a partial match between the new collection of object representations and the first collection of object representations. In further embodiments, the causing the logic circuit to receive the first one or more inputs correlated with the first collection of object representations includes transmitting, to the logic circuit, the first one or more inputs correlated with the first collection of object representations. In further embodiments, the causing the logic circuit to receive the first one or more inputs correlated with the first collection of object representations includes replacing one or more inputs into the logic circuit with the first one or more inputs correlated with the first collection of object representations.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first collection of object representations, the first collection of object representations including one or more representations of objects detected by a sensor. The operations may further comprise: receiving a first one or more inputs, wherein the first one or more inputs are also received

by a logic circuit, and wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device. The operations may further comprise: learning the first collection of object representations correlated with the first one or more inputs. The operations may further comprise: receiving a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The operations may further comprise: anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further comprise: causing the logic circuit to receive the first one or more inputs correlated with the first collection of object representations, the causing performed in response to the anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, wherein the device performs one or more operations defined by one or more outputs for operating the device produced by the logic circuit.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a processor circuit, the first collection of object representations including one or more representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more inputs by the processor circuit, wherein the first one or more inputs are also received by a logic circuit, and wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device. The method may further comprise: (c) learning the first collection of object representations correlated with the first one or more inputs, the learning of (c) performed by the processor circuit. The method may further comprise: (d) receiving a new collection of object representations by the processor circuit, the new collection of object representations including one or more representations of objects detected by the sensor. The method may further comprise: (e) anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (e) performed by the processor circuit. The method may further comprise: (f) receiving, by the logic circuit, the first one or more inputs correlated with the first collection of object representations, the receiving of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the device, one or more operations defined by one or more outputs for operating the device produced by the logic circuit.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for learning and using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a logic circuit configured to receive inputs and produce outputs, wherein the outputs are used for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence. The artificial intelligence unit may be configured to: receive a first collection of object representations, the first collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more outputs, the first one or more outputs transmitted from the logic circuit. The artificial intelligence unit may be further configured to: learn the first collection of object representations correlated with the first one or more outputs. The artificial intelligence unit may be further configured to: receive a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more outputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may be further configured to: cause the device to perform one or more operations defined by the first one or more outputs correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit.

In some embodiments, the learning the first collection of object representations correlated with the first one or more outputs includes correlating the first collection of object representations with the first one or more outputs. In further embodiments, the learning the first collection of object representations correlated with the first one or more outputs includes storing the first collection of object representations correlated with the first one or more outputs into the memory unit, the first collection of object representations correlated with the first one or more outputs being part of a plurality of collections of object representations correlated with one or more outputs stored in the memory unit. In further embodiments, the anticipating the first one or more outputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes comparing at least one portion of the new collection of object representations with at least one portion of the first collection of object representations. In further embodiments, the anticipating the first one or more outputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes determining that there is at least a partial match between the new collection of object representations and the first collection of object representations. In further embodiments, the causing the device to perform one or more operations defined by the first one or more outputs correlated with the first collection of object representations includes replacing one or more outputs from the logic circuit with the first one or more outputs correlated with the first collection of object representations.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first collection of object representations, the first collection of object representations including one or more representations of objects detected by a sensor. The opera-

tions may further comprise: receiving a first one or more outputs, the first one or more outputs transmitted from a logic circuit, wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device. The operations may further comprise: learning the first collection of object representations correlated with the first one or more outputs. The operations may further comprise: receiving a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The operations may further comprise: anticipating the first one or more outputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further comprise: causing the device to perform one or more operations defined by the first one or more outputs correlated with the first collection of object representations, the causing performed in response to the anticipating the first one or more outputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a processor circuit, the first collection of object representations including one or more representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more outputs by the processor circuit, the first one or more outputs transmitted from a logic circuit, wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device. The method may further comprise: (c) learning the first collection of object representations correlated with the first one or more outputs, the learning of (c) performed by the processor circuit. The method may further comprise: (d) receiving a new collection of object representations by the processor circuit, the new collection of object representations including one or more representations of objects detected by the sensor. The method may further comprise: (e) anticipating the first one or more outputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (e) performed by the processor circuit. The method may further comprise: (f) performing, by the device, one or more operations defined by the first one or more outputs correlated with the first collection of object representations, the one or more operations by the device performed in response to the anticipating of (e).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for learning and using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: an actuator configured to receive inputs and perform motions. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first collection of object representations, the first collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more inputs, wherein the first one or more inputs are also received by the actuator. The artificial intelligence unit may be further configured to: learn the first collection of object representations correlated with the first one or more inputs. The artificial intelligence unit may be further configured to: receive a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may be further configured to: cause the actuator to receive the first one or more inputs correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the actuator performs one or more motions defined by the first one or more inputs correlated with the first collection of object representations.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first collection of object representations, the first collection of object representations including one or more representations of objects detected by a sensor. The operations may further comprise: receiving a first one or more inputs, wherein the first one or more inputs are also received by an actuator, and wherein the actuator is configured to receive inputs and perform motions. The operations may further comprise: learning the first collection of object representations correlated with the first one or more inputs. The operations may further comprise: receiving a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The operations may further comprise: anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further comprise: causing the actuator to receive the first one or more inputs correlated with the first collection of object representations, the causing performed in response to the anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, wherein the actuator performs one or more motions defined by the first one or more inputs correlated with the first collection of object representations.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a processor circuit, the first collection of object representations including one or more representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more inputs by the processor circuit, wherein the first one or more inputs are also received by an actuator, and wherein the actuator is

configured to receive inputs and perform motions. The method may further comprise: (c) learning the first collection of object representations correlated with the first one or more inputs, the learning of (c) performed by the processor circuit. The method may further comprise: (d) receiving a new collection of object representations by the processor circuit, the new collection of object representations including one or more representations of objects detected by the sensor. The method may further comprise: (e) anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (e) performed by the processor circuit. The method may further comprise: (f) receiving, by the actuator, the first one or more inputs correlated with the first collection of object representations, the receiving of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the actuator, one or more motions defined by the first one or more inputs correlated with the first collection of object representations.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

Other features and advantages of the disclosure will become apparent from the following description, including the claims and drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** illustrates a block diagram of Computing Device **70** that can provide processing capabilities used in some of the disclosed embodiments.

FIG. **2** illustrates an embodiment of Device **98** comprising Unit for Learning and/or Using a Device's Circumstances for Autonomous Device Operation (DCADO Unit **100**).

FIGS. **3**A-**3**E illustrate various embodiments of Sensors **92** and elements of Object Processing Unit **93**.

FIGS. **4**A-**4**B, illustrate an exemplary embodiment of Objects **615** detected in Device's **98** surrounding, and resulting Collection of Object Representations **525**.

FIG. **5** illustrates some embodiments of obtaining instruction sets, data, and/or other information through tracing, profiling, or sampling of Processor **11** registers, memory, or other computing system components.

FIGS. **6**A-**6**B illustrate some embodiments of obtaining instruction sets, data, and/or other information through tracing, profiling, or sampling of Logic Circuit **250**.

FIGS. **7**A-**7**E illustrate some embodiments of Instruction Sets **526**.

FIGS. **8**A-**8**B illustrate some embodiments of Extra Information **527**.

FIG. **9** illustrates an embodiment where DCADO Unit **100** is part of or operating on Processor **11**.

FIG. **10** illustrates an embodiment where DCADO Unit **100** resides on Server **96** accessible over Network **95**.

FIG. **11** illustrates an embodiment of learning and/or using Remote Device's **97** circumstances for autonomous Remote Device **97** operation.

FIG. **12** illustrates an embodiment of Artificial Intelligence Unit **110**.

FIG. **13** illustrates an embodiment of Knowledge Structuring Unit **520** correlating individual Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527**.

FIG. **14** illustrates another embodiment of Knowledge Structuring Unit **520** correlating individual Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527**.

FIG. **15** illustrates an embodiment of Knowledge Structuring Unit **520** correlating streams of Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527**.

FIG. **16** illustrates another embodiment of Knowledge Structuring Unit **520** correlating streams of Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527**.

FIG. **17** illustrates various artificial intelligence methods, systems, and/or models that can be utilized in DCADO Unit **100** embodiments.

FIG. **18**A-**18**C illustrate embodiments of interconnected Knowledge Cells **800** and updating weights of Connections **853**.

FIG. **19** illustrates an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Collection of Knowledge Cells **530***d*.

FIG. **20** illustrates an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Neural Network **530***a*.

FIG. **21** illustrates an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Neural Network **530***a* comprising shortcut Connections **853**.

FIG. **22** illustrates an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Graph **530***b*.

FIG. **23** illustrates an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Collection of Sequences **530***c*.

FIG. **24** illustrates an embodiment of determining anticipatory Instruction Sets **526** from a single Knowledge Cell **800**.

FIG. **25** illustrates an embodiment of determining anticipatory Instruction Sets **526** by traversing a single Knowledge Cell **800**.

FIG. **26** illustrates an embodiment of determining anticipatory Instruction Sets **526** using collective similarity comparisons.

FIG. **27** illustrates an embodiment of determining anticipatory Instruction Sets **526** using Neural Network **530***a*.

FIG. **28** illustrates an embodiment of determining anticipatory Instruction Sets **526** using Graph **530***b*.

FIG. **29** illustrates an embodiment of determining anticipatory Instruction Sets **526** using Collection of Sequences **530***c*.

FIG. **30** illustrates some embodiments of modifying execution and/or functionality of Processor **11** through modification of Processor **11** registers, memory, or other computing system components.

FIGS. **31**A-**31**B illustrate some embodiments of modifying execution and/or functionality of Logic Circuit **250** through modification of inputs and/or outputs of Logic Circuit **250**.

FIG. **32** illustrates a flow chart diagram of an embodiment of method **9100** for learning and/or using a device's circumstances for autonomous device operation.

FIG. **33** illustrates a flow chart diagram of an embodiment of method **9200** for learning and/or using a device's circumstances for autonomous device operation.

FIG. **34** illustrates a flow chart diagram of an embodiment of method **9300** for learning and/or using a device's circumstances for autonomous device operation.

FIG. **35** illustrates a flow chart diagram of an embodiment of method **9400** for learning and/or using a device's circumstances for autonomous device operation.

FIG. **36** illustrates a flow chart diagram of an embodiment of method **9500** for learning and/or using a device's circumstances for autonomous device operation.

FIG. **37** illustrates a flow chart diagram of an embodiment of method **9600** for learning and/or using a device's circumstances for autonomous device operation.

FIG. **38** illustrates an exemplary embodiment of Loader **98***a*.

FIG. **39** illustrates an exemplary embodiment of Boat **98***b*.

FIG. **40** illustrates an exemplary embodiment of utilizing Area of Interest **450** around Boat **98***b*.

Like reference numerals in different figures indicate like elements. Horizontal or vertical "..." or other such indicia may be used to indicate additional instances of the same type of element. n, m, x, or other such letters or indicia represent integers or other sequential numbers that follow the sequence where they are indicated. It should be noted that n, m, x, or other such letters or indicia may represent different numbers in different elements even where the elements are depicted in the same figure. In general, n, m, x, or other such letters or indicia may follow the sequence and/or context where they are indicated. Any of these or other such letters or indicia may be used interchangeably depending on context and space available. The drawings are not necessarily to scale, with emphasis instead being placed upon illustrating the embodiments, principles, and concepts of the disclosure. A line or arrow between any of the disclosed elements comprises an interface that enables the coupling, connection, and/or interaction between the elements.

DETAILED DESCRIPTION

The disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation comprise apparatuses, systems, methods, features, functionalities, and/or applications that enable learning a device's circumstances including objects with various properties along with correlated instruction sets for operating the device, storing this knowledge in a knowledgebase (i.e. neural network, graph, sequences, etc.), and operating a device autonomously. The disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, any of their elements, any of their embodiments, or a combination thereof can generally be referred to as DCADO, DCADO Unit, or as other suitable name or reference.

Referring now to FIG. **1**, an embodiment is illustrated of Computing Device **70** (also referred to simply as computing device, computing system, or other suitable name or reference, etc.) that can provide processing capabilities used in some embodiments of the forthcoming disclosure. Later described devices, systems, and methods, in combination with processing capabilities of Computing Device **70**, enable learning and/or using a device's circumstances for autonomous device operation and/or other functionalities described herein. Various embodiments of the disclosed devices, systems, and methods include hardware, functions, logic, programs, and/or a combination thereof that can be implemented using any type or form of computing, computing enabled, or other device or system such as a mobile device, a computer, a computing enabled telephone, a server, a gaming device, a television device, a digital camera, a GPS receiver, a media player, an embedded device, a supercomputer, a wearable device, an implantable device, a cloud, or any other type or form of computing, computing enabled, or other device or system capable of performing the operations described herein.

In some designs, Computing Device **70** comprises hardware, processing techniques or capabilities, programs, or a combination thereof. Computing Device **70** includes one or more central processing units, which may also be referred to as processors **11**. Processor **11** includes one or more memory ports **10** and/or one or more input-output ports, also referred to as I/O ports **15**, such as I/O ports **15**A and **15**B. Processor **11** may be special or general purpose. Computing Device **70** may further include memory **12**, which can be connected to the remainder of the components of Computing Device **70** via bus **5**. Memory **12** can be connected to processor **11** via memory port **10**. Computing Device **70** may also include display device **21** such as a monitor, projector, glasses, and/or other display device. Computing Device **70** may also include Human-machine Interface **23** such as a keyboard, a pointing device, a mouse, a touchscreen, a joystick, a remote controller, and/or other input device. In some implementations, Human-machine Interface **23** can be connected with bus **5** or directly connected with specific elements of Computing Device **70**. Computing Device **70** may include additional elements such as one or more input/output devices **13**. Processor **11** may include or be interfaced with cache memory **14**. Storage **27** may include memory, which provides an operating system **17** (i.e. also referred to as OS **17**, etc.), additional application programs **18**, and/or data space **19** in which additional data or information can be stored. Alternative memory device **16** can be connected to the remaining components of Computing Device **70** via bus **5**. Network interface **25** can also be connected with bus **5** and be used to communicate with external computing devices via a network. Some or all described elements of Computing Device **70** can be directly or operatively connected or coupled with each other using any other connection means known in art. Other additional elements may be included as needed, or some of the disclosed ones may be excluded, or a combination thereof may be utilized in alternate implementations of Computing Device **70**.

Processor **11** includes one or more circuits or devices that can execute instructions fetched from memory **12** and/or other element. Processor **11** may include any combination of hardware and/or processing techniques or capabilities for executing or implementing logic functions or programs. Processor **11** may include a single core or a multi core processor. Processor **11** includes the functionality for loading operating system **17** and operating any application programs **18** thereon. In some embodiments, Processor **11** can be provided in a microprocessing or a processing unit, such as, for example, Snapdragon processor produced by Qualcomm Inc., processor by Intel Corporation of Mountain View, Calif., processor manufactured by Motorola Corpo-

ration of Schaumburg, Ill.; processor manufactured by Transmeta Corporation of Santa Clara, Calif.; processor manufactured by International Business Machines of White Plains, N.Y.; processor manufactured by Advanced Micro Devices of Sunnyvale, Calif., or any computing circuit or device for performing similar functions. In other embodiments, processor **11** can be provided in a graphics processing unit (GPU), visual processing unit (VPU), or other highly parallel processing circuit or device such as, for example, nVidia GeForce line of GPUs, AMD Radeon line of GPUs, and/or others. Such GPUs or other highly parallel processing circuits or devices may provide superior performance in processing operations on neural networks, graphs, and/or other data structures. In further embodiments, processor **11** can be provided in a micro controller such as, for example, Texas instruments, Atmel, Microchip Technology, ARM, Silicon Labs, Intel, and/or other lines of micro controllers. In further embodiments, processor **11** can be provided in a quantum processor such as, for example, D-Wave Systems, Microsoft, Intel, IBM, Google, Toshiba, and/or other lines of quantum processors. In further embodiments, processor **11** can be provided in a biocomputer such as DNA-based computer, protein-based computer, molecule-based computer, and/or others. In further embodiments, processor **11** includes any circuit or device for performing logic operations. Processor **11** can be based on any of the aforementioned or other available processors capable of operating as described herein. Computing Device **70** may include one or more of the aforementioned or other processors. In some designs, processor **11** can communicate with memory **12** via a system bus **5**. In other designs, processor **11** can communicate directly with memory **12** via a memory port **10**.

Memory **12** includes one or more circuits or devices capable of storing data. In some embodiments, Memory **12** can be provided in a semiconductor or electronic memory chip such as static random access memory (SRAM), Flash memory, Burst SRAM or SynchBurst SRAM (BSRAM), Dynamic random access memory (DRAM), Fast Page Mode DRAM (FPM DRAM), Enhanced DRAM (EDRAM), Extended Data Output RAM (EDO RAM), Extended Data Output DRAM (EDO DRAM), Burst Extended Data Output DRAM (BEDO DRAM), Enhanced DRAM (EDRAM), synchronous DRAM (SDRAM), JEDEC SRAM, PC100 SDRAM, Double Data Rate SDRAM (DDR SDRAM), Enhanced SDRAM (ESDRAM), SyncLink DRAM (SLDRAM), Direct Rambus DRAM (DRDRAM), Ferroelectric RAM (FRAM), and/or others. In other embodiments, Memory **12** includes any volatile memory. In general, Memory **12** can be based on any of the aforementioned or other available memories capable of operating as described herein.

Storage **27** includes one or more devices or mediums capable of storing data. In some embodiments, Storage **27** can be provided in a device or medium such as a hard drive, flash drive, optical disk, and/or others. In other embodiments, Storage **27** can be provided in a biological storage device such as DNA-based storage device, protein-based storage device, molecule-based storage device, and/or others. In further embodiments, Storage **27** can be provided in an optical storage device such as holographic storage, and/or others. In further embodiments, Storage **27** may include any non-volatile memory. In general, Storage **27** can be based on any of the aforementioned or other available storage devices or mediums capable of operating as described herein. In some aspects, Storage **27** may include any features, functionalities, and embodiments of Memory **12**, and vice versa, as applicable.

Processor **11** can communicate directly with cache memory **14** via a connection means such as a secondary bus which may also sometimes be referred to as a backside bus. In some embodiments, processor **11** can communicate with cache memory **14** using the system bus **5**. Cache memory **14** may typically have a faster response time than main memory **12** and can include a type of memory which is considered faster than main memory **12** such as, for example, SRAM, BSRAM, or EDRAM. Cache memory includes any structure such as multilevel caches, for example. In some embodiments, processor **11** can communicate with one or more I/O devices **13** via a system bus **5**. Various busses can be used to connect processor **11** to any of the I/O devices **13** such as a VESA VL bus, an ISA bus, an EISA bus, a MicroChannel Architecture (MCA) bus, a PCI bus, a PCI-X bus, a PCI-Express bus, a NuBus, and/or others. In some embodiments, processor **11** can communicate directly with I/O device **13** via HyperTransport, Rapid I/O, or InfiniBand. In further embodiments, local busses and direct communication can be mixed. For example, processor **11** can communicate with an I/O device **13** using a local interconnect bus and communicate with another I/O device **13** directly. Similar configurations can be used for any other components described herein.

Computing Device **70** may further include alternative memory such as a SD memory slot, a USB memory stick, an optical drive such as a CD-ROM drive, a CD-R/RW drive, a DVD-ROM drive or a BlueRay disc, a hard-drive, and/or any other device comprising non-volatile memory suitable for storing data or installing application programs. Computing Device **70** may further include a storage device **27** comprising any type or form of non-volatile memory for storing an operating system (OS) such as any type or form of Windows OS, Mac OS, Unix OS, Linux OS, Android OS, iPhone OS, mobile version of Windows OS, an embedded OS, or any other OS that can operate on Computing Device **70**. Computing Device **70** may also include application programs **18**, and/or data space **19** for storing additional data or information. In some embodiments, alternative memory **16** can be used as or similar to storage device **27**. Additionally, OS **17** and/or application programs **18** can be operable from a bootable medium such as, for example, a flash drive, a micro SD card, a bootable CD or DVD, and/or other bootable medium.

Application Program **18** (also referred to as program, computer program, application, script, code, or other suitable name or reference) comprises instructions that can provide functionality when executed by processor **11**. As such, Application Program **18** may be used to operate (i.e. perform operations on/with) or control a device or system. Application program **18** can be implemented in a high-level procedural or object-oriented programming language, or in a low-level machine or assembly language. Any language used can be compiled, interpreted, or otherwise translated into machine language. Application program **18** can be deployed in any form including as a stand-alone program or as a module, component, subroutine, or other unit suitable for use in a computing system. Application program **18** does not necessarily correspond to a file in a file system. A program can be stored in a portion of a file that may hold other programs or data, in a single file dedicated to the program, or in multiple files (i.e. files that store one or more modules, sub programs, or portions of code, etc.). Application Program **18** can be delivered in various forms such as, for example, executable file, library, script, plugin, addon, applet, interface, console application, web application, application service provider (ASP)-type application, operating system, and/or other forms. Application program **18**

can be deployed to be executed on one computing device or on multiple computing devices (i.e. cloud, distributed, or parallel computing, etc.), or at one site or distributed across multiple sites interconnected by a communication network or an interface.

Network interface **25** can be utilized for interfacing Computing Device **70** with other devices via a network through a variety of connections including telephone lines, wired or wireless connections, LAN or WAN links (i.e. 802.11, T1, T3, 56 kb, X.25, etc.), broadband connections (i.e. ISDN, Frame Relay, ATM, etc.), or a combination thereof. Examples of networks include the Internet, an intranet, an extranet, a local area network (LAN), a wide area network (WAN), a personal area network (PAN), a home area network (HAN), a campus area network (CAN), a metropolitan area network (MAN), a global area network (GAN), a storage area network (SAN), virtual network, a virtual private network (VPN), a Bluetooth network, a wireless network, a wireless LAN, a radio network, a HomePNA, a power line communication network, a G.hn network, an optical fiber network, an Ethernet network, an active networking network, a client-server network, a peer-to-peer network, a bus network, a star network, a ring network, a mesh network, a star-bus network, a tree network, a hierarchical topology network, and/or other networks. Network interface **25** may include a built-in network adapter, network interface card, PCMCIA network card, card bus network adapter, wireless network adapter, Bluetooth network adapter, WiFi network adapter, USB network adapter, modem, and/or any other device suitable for interfacing Computing Device **70** with any type of network capable of communication and/or operations described herein.

I/O devices **13** may be present in various shapes or forms in Computing Device **70**. Examples of I/O device **13** capable of input include a joystick, a keyboard, a mouse, a trackpad, a trackpoint, a touchscreen, a trackball, a microphone, a drawing tablet, a glove, a tactile input device, a still or video camera, and/or other input device. Examples of I/O device **13** capable of output include a video display, a touchscreen, a projector, a glasses, a speaker, a tactile output device, and/or other output device. Examples of I/O device **13** capable of input and output include a disk drive, an optical storage device, a modem, a network card, and/or other input/output device. I/O device **13** can be interfaced with processor **11** via an I/O port **15**, for example. In some aspects, I/O device **13** can be a bridge between system bus **5** and an external communication bus such as a USB bus, an Apple Desktop Bus, an RS-232 serial connection, a SCSI bus, a FireWire bus, a FireWire 800 bus, an Ethernet bus, an AppleTalk bus, a Gigabit Ethernet bus, an Asynchronous Transfer Mode bus, a HIPPI bus, a Super HIPPI bus, a SerialPlus bus, a SCI/LAMP bus, a FibreChannel bus, a Serial Attached small computer system interface bus, and/or other bus.

An output interface (not shown) such as a graphical user interface, an acoustic output interface, a tactile output interface, any device driver (i.e. audio, video, or other driver), and/or other output interface or system can be utilized to process output from elements of Computing Device **70** for conveyance on an output device such as Display **21**. In some aspects, Display **21** or other output device itself may include an output interface for processing output from elements of Computing Device **70**. Further, an input interface (not shown) such as a keyboard listener, a touchscreen listener, a mouse listener, any device driver (i.e. audio, video, keyboard, mouse, touchscreen, or other driver), and/or other input interface or system can be utilized to process input from Human-machine Interface **23** or other input device for use by elements of Computing Device **70**. In some aspects, Human-machine Interface **23** or other input device itself may include an input interface for processing input for use by elements of Computing Device **70**.

Computing Device **70** may include or be connected to multiple display devices **21**. Display devices **21** can each be of the same or different type or form. Computing Device **70** and/or its elements comprise any type or form of suitable hardware, programs, or a combination thereof to support, enable, or provide for the connection and use of multiple display devices **21**. In one example, Computing Device **70** includes any type or form of video adapter, video card, driver, and/or library to interface, communicate, connect, or otherwise use display devices **21**. In some aspects, a video adapter may include multiple connectors to interface to multiple display devices **21**. In other aspects, Computing Device **70** includes multiple video adapters, with each video adapter connected to one or more display devices **21**. In some embodiments, Computing Device's **70** operating system can be configured for using multiple displays **21**. In other embodiments, one or more display devices **21** can be provided by one or more other computing devices such as remote computing devices connected to Computing Device **70** via a network or an interface.

Computing Device **70** can operate under the control of operating system **17**, which may support Computing Device's **70** basic functions, interface with and manage hardware resources, interface with and manage peripherals, provide common services for application programs, schedule tasks, and/or perform other functionalities. A modern operating system enables features and functionalities such as a high resolution display, graphical user interface (GUI), touchscreen, cellular network connectivity (i.e. mobile operating system, etc.), Bluetooth connectivity, WiFi connectivity, global positioning system (GPS) capabilities, mobile navigation, microphone, speaker, still picture camera, video camera, voice recorder, speech recognition, music player, video player, near field communication, personal digital assistant (PDA), and/or other features, functionalities, or applications. For example, Computing Device **70** can use any conventional operating system, any embedded operating system, any real-time operating system, any open source operating system, any video gaming operating system, any proprietary operating system, any online operating system, any operating system for mobile computing devices, or any other operating system capable of running on Computing Device **70** and performing operations described herein. Example of operating systems include Windows XP, Windows 7, Windows 8, Windows 10, etc. manufactured by Microsoft Corporation of Redmond, Wash.; Mac OS, iPhone OS, etc. manufactured by Apple Computer of Cupertino, Calif.; OS/2 manufactured by International Business Machines of Armonk, N.Y.; Linux, a freely-available operating system distributed by Caldera Corp. of Salt Lake City, Utah; or any type or form of a Unix operating system, and/or others. Any operating systems such as the ones for Android devices can similarly be utilized.

Computing Device **70** can be implemented as or be part of various model architectures such as web services, distributed computing, grid computing, cloud computing, and/or other architectures. For example, in addition to the traditional desktop, server, or mobile operating system architectures, a cloud-based operating system can be utilized to provide the structure on which embodiments of the disclosure can be implemented. Other aspects of Computing

Device **70** can also be implemented in the cloud without departing from the spirit and scope of the disclosure. For example, memory, storage, processing, and/or other elements can be hosted in the cloud. In some embodiments, Computing Device **70** can be implemented on multiple devices. For example, a portion of Computing Device **70** can be implemented on a mobile device and another portion can be implemented on wearable electronics.

Computing Device **70** can be or include any mobile device, a mobile phone, a smartphone (i.e. iPhone, Windows phone, Blackberry phone, Android phone, etc.), a tablet, a personal digital assistant (PDA), wearable electronics, implantable electronics, and/or other mobile device capable of implementing the functionalities described herein. Computing Device **70** can also be or include an embedded device, which can be any device or system with a dedicated function within another device or system. Embedded systems range from the simplest ones dedicated to one task with no user interface to complex ones with advanced user interface that may resemble modern desktop computer systems. Examples of devices comprising an embedded device include a mobile telephone, a personal digital assistant (PDA), a gaming device, a media player, a digital still or video camera, a pager, a television device, a set-top box, a personal navigation device, a global positioning system (GPS) receiver, a portable storage device (i.e. a USB flash drive, etc.), a digital watch, a DVD player, a printer, a microwave oven, a washing machine, a dishwasher, a gateway, a router, a hub, an automobile entertainment system, an automobile navigation system, a refrigerator, a washing machine, a factory automation device, an assembly line device, a factory floor monitoring device, a thermostat, an automobile, a factory controller, a telephone, a network bridge, and/or other devices. An embedded device can operate under the control of an operating system for embedded devices such as MicroC/OS-II, QNX, VxWorks, eCos, TinyOS, Windows Embedded, Embedded Linux, and/or other embedded device operating systems.

Various implementations of the disclosed devices, systems, and methods can be realized in digital electronic circuitry, integrated circuitry, logic gates, application specific integrated circuits (ASICs), field programmable gate arrays (FPGAs), computer hardware, firmware, programs, virtual machines, and/or combinations thereof including their structural, logical, and/or physical equivalents.

The disclosed devices, systems, and methods may include clients and servers. A client and server are generally, but not always, remote from each other and typically, but not always, interact via a network or an interface. The relationship of a client and server may arise by virtue of computer programs running on their respective computers and having a client-server relationship to each other, for example.

The disclosed devices, systems, and methods can be implemented in a computing system that includes a back end component, a middleware component, a front end component, or any combination thereof. The components of the system can be interconnected by any form or medium of digital data communication such as, for example, a network.

Computing Device **70** may include or be interfaced with a computer program product comprising instructions or logic encoded on a computer-readable medium. Such instructions or logic, when executed, may configure or cause one or more processors to perform the operations and/or functionalities disclosed herein. For example, a computer program can be provided or encoded on a computer-readable medium such as an optical medium (i.e. DVD-ROM, etc.), flash drive, hard drive, any memory, firmware, or other medium. Computer program can be installed onto a computing device to cause the computing device to perform the operations and/or functionalities disclosed herein. Machine-readable medium, computer-readable medium, or other such terms may refer to any computer program product, apparatus, and/or device for providing instructions and/or data to one or more programmable processors. As such, machine-readable medium includes any medium that can send and/or receive machine instructions as a machine-readable signal. Examples of a machine-readable medium include a volatile and/or non-volatile medium, a removable and/or non-removable medium, a communication medium, a storage medium, and/or other medium. A communication medium, for example, can transmit computer readable instructions and/or data in a modulated data signal such as a carrier wave or other transport technique, and may include any other form of information delivery medium known in art. A non-transitory machine-readable medium comprises all machine-readable media except for a transitory, propagating signal.

In some embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, can be implemented entirely or in part in a device (i.e. microchip, circuitry, logic gates, electronic device, computing device, special or general purpose processor, etc.) or system that comprises (i.e. hard coded, internally stored, etc.) or is provided with (i.e. externally stored, etc.) instructions for implementing DCADO functionalities. As such, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, may include the processing, memory, storage, and/or other features, functionalities, and embodiments of Computing Device **70** or elements thereof. Such device or system can operate on its own (i.e. standalone device or system, etc.), be embedded in another device or system (i.e. an industrial machine, a robot, a vehicle, a toy, a smartphone, a television device, an appliance, and/or any other device or system capable of housing the elements needed for DCADO functionalities), work in combination with other devices or systems, or be available in any other configuration. In other embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, may include Alternative Memory **16** that provides instructions for implementing DCADO functionalities to one or more Processors **11**. In further embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, can be implemented entirely or in part as a computer program and executed by one or more Processors **11**. Such program can be implemented in one or more modules or units of a single or multiple computer programs. Such program may be able to attach to or interface with, inspect, and/or take control of another application program to implement DCADO functionalities. In further embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, can be implemented as a network, web, distributed, cloud, or other such application accessed on one or more remote computing devices (i.e. servers, cloud, etc.) via Network Interface **25**, such remote computing devices including processing capabilities and instructions for implementing DCADO functionalities. In further embodiments, the disclosed artificially intelligent devices, systems, and

methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, can be (1) attached to or interfaced with any computing device or application program, (2) included as a feature of an operating system, (3) built (i.e. hard coded, etc.) into any computing device or application program, and/or (4) available in any other configuration to provide its functionalities.

In some embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, can be implemented at least in part in a computer program such as Java application or program. Java provides a robust and flexible environment for application programs including flexible user interfaces, robust security, built-in network protocols, powerful application programming interfaces, database or DBMS connectivity and interfacing functionalities, file manipulation capabilities, support for networked applications, and/or other features or functionalities. Application programs based on Java can be portable across many devices, yet leverage each device's native capabilities. Java supports the feature sets of most smartphones and a broad range of connected devices while still fitting within their resource constraints. Various Java platforms include virtual machine features comprising a runtime environment for application programs. Java platforms provide a wide range of user-level functionalities that can be implemented in application programs such as displaying text and graphics, playing and recording audio content, displaying and recording visual content, communicating with another computing device, and/or other functionalities. It should be understood that the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, are programming language, platform, and operating system independent. Examples of programming languages that can be used instead of or in addition to Java include C, C++, Cobol, Python, Java Script, Tcl, Visual Basic, Pascal, VB Script, Perl, PHP, Ruby, and/or other programming languages capable of implementing the functionalities described herein.

Where a reference to a specific file or file type is used herein, other files or file types can be substituted.

Where a reference to a data structure is used herein, it should be understood that any variety of data structures can be used such as, for example, array, list, linked list, doubly linked list, queue, tree, heap, graph, map, grid, matrix, multi-dimensional matrix, table, database, database management system (DBMS), file, neural network, and/or any other type or form of a data structure including a custom one. A data structure may include one or more fields or data fields that are part of or associated with the data structure. A field or data field may include a data, an object, a data structure, and/or any other element or a reference/pointer thereto. A data structure can be stored in one or more memories, files, or other repositories. A data structure and/or elements thereof, when stored in a memory, file, or other repository, may be stored in a different arrangement than the arrangement of the data structure and/or elements thereof. For example, a sequence of elements can be stored in an arrangement other than a sequence in a memory, file, or other repository.

Where a reference to a repository is used herein, it should be understood that a repository may be or include one or more files or file systems, one or more storage locations or structures, one or more storage systems, one or more memory locations or structures, and/or other file, storage, memory, or data arrangements.

Where a reference to an interface is used herein, it should be understood that the interface comprises any hardware, device, system, program, method, and/or combination thereof that enable direct or operative coupling, connection, and/or interaction of the elements between which the interface is indicated. A line or arrow shown in the figures between any of the depicted elements comprises such interface. Examples of an interface include a direct connection, an operative connection, a wired connection (i.e. wire, cable, etc.), a wireless connection, a device, a network, a bus, a circuit, a firmware, a driver, a bridge, a program, a combination thereof, and/or others.

Where a reference to an element coupled or connected to another element is used herein, it should be understood that the element may be in communication or other interactive relationship with the other element. Furthermore, an element coupled or connected to another element can be coupled or connected to any other element in alternate implementations. Terms coupled, connected, interfaced, or other such terms may be used interchangeably herein depending on context.

Where a reference to an element matching another element is used herein, it should be understood that the element may be equivalent or similar to the other element. Therefore, the term match or matching can refer to total equivalence or similarity depending on context.

Where a reference to a device is used herein, it should be understood that the device may include or be referred to as a system, and vice versa depending on context, since a device may include a system of elements and a system may be embodied in a device.

Where a reference to a collection of elements is used herein, it should be understood that the collection of elements may include one or more elements. In some aspects or contexts, a reference to a collection of elements does not imply that the collection is an element itself.

Where a reference to an object is used herein, it should be understood that the object may be a physical object (i.e. object detected in a device's surrounding, etc.), an electronic object (i.e. object in an object oriented application program, etc.), and/or other object depending on context.

Where a mention of a function, method, routine, subroutine, or other such procedure is used herein, it should be understood that the function, method, routine, subroutine, or other such procedure comprises a call, reference, or pointer to the function, method, routine, subroutine, or other such procedure.

Where a mention of data, object, data structure, item, element, or thing is used herein, it should be understood that the data, object, data structure, item, element, or thing comprises a reference or pointer to the data, object, data structure, item, element, or thing.

Referring to FIG. **2**, an embodiment of Device **98** comprising Unit for Learning and/or Using a Device's Circumstances for Autonomous Device Operation (DCADO Unit **100**) is illustrated. Device **98** also comprises interconnected Processor **11**, Human-machine Interface **23**, Sensor **92**, Object Processing Unit **93**, Memory **12**, and Storage **27**. Processor **11** includes or executes Application Program **18**. DCADO Unit **100** comprises interconnected Artificial Intelligence Unit **110**, Acquisition Interface **120**, and Modification Interface **130**. Other additional elements can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate embodiments.

In one example, the teaching presented by the disclosure can be implemented in a device or system for learning and/or using a device's circumstances for autonomous device operation. The device or system may include a processor circuit (i.e. Processor **11**, etc.) configured to execute instruction sets (i.e. Instruction Sets **526**, etc.) for operating a device. The device or system may further include a memory unit (i.e. Memory **12**, etc.) configured to store data. The device or system may further include a sensor (i.e. Sensor **92**, etc.) configured to detect objects (i.e. Objects **615**, etc.). The device or system may further include an artificial intelligence unit (i.e. Artificial Intelligence Unit **110**, etc.). The artificial intelligence unit may be configured to receive a first collection of object representations (i.e. Collection of Object Representations **525**, etc.), the first collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may also be configured to receive a first one or more instruction sets for operating the device. The artificial intelligence unit may also be configured to learn the first collection of object representations correlated with the first one or more instruction sets for operating the device. The artificial intelligence unit may also be configured to receive a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may also be configured to anticipate the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may also be configured to cause the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations performed in response to the executing by the processor circuit. Any of the operations of the aforementioned elements can be performed repeatedly and/or in different orders in alternate embodiments. In some embodiments, a stream of collections of object representations can be used instead of or in addition to any collection of object representations such as, for example, using a first stream of collections of object representations instead of the first collection of object representations. In other embodiments, a logic circuit (i.e. Logic Circuit **250**, etc.) may be used instead of the processor circuit. In such embodiments, one or more instruction sets for operating the device (i.e. first one or more instruction sets for operating the device, etc.) may include or be substituted with one or more inputs into or one or more outputs from the logic circuit. In further embodiments, an actuator may be included instead of or in addition to the processor circuit. In such embodiments, one or more instruction sets for operating the device (i.e. first one or more instruction sets for operating the device, etc.) may include or be substituted with one or more inputs into the actuator. Other additional elements can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate embodiments. The device or system for learning and/or using a device's circumstances for autonomous device operation may include any actions or operations of any of the disclosed methods such as methods **9100**, **9200**, **9300**, **9400**, **9500**, **9600**, and/or others (all later described).

Device **98** comprises any hardware, programs, or a combination thereof. Although, Device **98** is referred to as a device herein, Device **98** may be or include a system as a system may be embodied in Device **98**. Device **98** may include any features, functionalities, and embodiments of Computing Device **70**, or elements thereof. In some embodiments, Device **98** includes a computing enabled device for performing mechanical or physical operations (i.e. via actuators, etc.). In other embodiments, Device **98** includes a computing enabled device for performing non-mechanical and/or other operations. Examples of Device **98** include an industrial machine, a toy, a robot, a vehicle, an appliance, a control device, a smartphone or other mobile computer, any computer, and/or other computing enabled device or machine. Such device or machine may be built for any function or purpose some examples of which are described later.

User **50** (also referred to simply as user or other suitable name or reference) comprises a human user or non-human user. A non-human User **50** includes any device, system, program, and/or other mechanism for operating or controlling Device **98** and/or elements thereof. In one example, User **50** may issue an operating direction to Application Program **18** responsive to which Application Program's **18** instructions or instruction sets may be executed by Processor **11** to perform a desired operation on Device **98**. In another example, User **50** may issue an operating direction to Processor **11**, Logic Circuit **250** (later described), and/or other processing element responsive to which Processor **11**, Logic Circuit **250**, and/or other processing element may implement logic to perform a desired operation on Device **98**. User's **50** operating directions comprise any user inputted data (i.e. values, text, symbols, etc.), directions (i.e. move right, move up, move forward, copy an item, click on a link, etc.), instructions or instruction sets (i.e. manually inputted instructions or instruction sets, etc.), and/or other inputs or information. A non-human User **50** can utilize more suitable interfaces instead of, or in addition to, Human-machine Interface **23** and/or Display **21** for controlling Device **98** and/or elements thereof. Examples of such interfaces include an application programming interface (API), bridge (i.e. bridge between applications, devices, or systems, etc.), driver, socket, direct or operative connection, handle, function/routine/subroutine, and/or other interfaces.

In some embodiments, Processor **11**, Logic Circuit **250**, Application Program **18**, and/or other processing element may control or affect an actuator (not shown). Actuator comprises the functionality for implementing motion, actions, behaviors, maneuvers, and/or other mechanical or physical operations. Device **98** may include one or more actuators to enable Device **98** to perform mechanical, physical, or other operations and/or to interact with its environment. For example, an actuator may include or be coupled to an element such as a wheel, arm, or other element to act upon the environment. Examples of an actuator include a motor, a linear motor, a servomotor, a hydraulic element, a pneumatic element, an electro-magnetic element, a spring element, and/or other actuators. Examples of types of actuators include a rotary actuator, a linear actuator, and/or other types of actuators. In other embodiments, Processor **11**, Logic Circuit **250**, Application Program **18**, and/or other processing element may control or affect any other device or element instead of or in addition to an actuator.

Referring to FIGS. **3**A-**3**E, various embodiments of Sensors **92** and elements of Object Processing Unit **93** are illustrated.

Sensor **92** (also referred to simply as sensor or other suitable name or reference) comprises the functionality for obtaining or detecting information about its environment, and/or other functionalities. As such, one or more Sensors **92** can be used to detect objects and/or their properties in Device's **98** surrounding. In some aspects, Device's **98** surrounding may include exterior of Device **98**. In other aspects, Device's **98** surrounding may include interior of Device **98** in case of hollow Device **98**, Device **98** comprising compartments or openings, and/or other variously shaped Device **98**. Examples of aspects of an environment that Sensor **92** can measure or be sensitive to include light (i.e. camera, lidar, etc.), electromagnetism/electromagnetic field (i.e. radar, etc.), sound (i.e. microphone, sonar, etc.), physical contact (i.e. tactile sensor, etc.), magnetism/magnetic field (i.e. compass, etc.), electricity/electric field, temperature, gravity, vibration, pressure, and/or others. In some aspects, a passive sensor (i.e. camera, microphone, etc.) measures signals or radiation emitted or reflected by an object. In other aspects, an active sensor (i.e. lidar, radar, sonar, etc.) emits signals or radiation and measures the signals or radiation reflected or backscattered from an object. A reference to a Sensor **92** herein includes a reference to one or more Sensors **92** as applicable. In some designs, a plurality of Sensors **92** may be used to detect objects and/or their properties from different angles or sides of Device **98**. For example, four Cameras **92***a* can be placed on four corners of Device **98** to cover 360 degrees of view of Device's **98** surrounding. In other designs, a plurality of different types of Sensors **92** may be used to detect different types of objects and/or their properties. For example, one or more Cameras **92***a* can be used to detect and identify an object, whereas, Radar **92***d* can be used to determine distance and bearing/angle of the object relative to Device **98**. In further designs, a signal-emitting element can be placed within or onto an object and Sensor **92** can detect the signal from the signal-emitting element, thereby detecting the object and/or its properties. For example, a radio-frequency identification (RFID) emitter may be placed within an object to help Sensor **92** detect, identify, and/or obtain other information about the object.

In some embodiments, Sensor **92** may be or include Camera **92***a* as shown in FIG. **3**A. Camera **92***a* comprises the functionality for capturing one or more pictures, and/or other functionalities. As such, Camera **92***a* can be used to capture pictures of Device's **98** surrounding. Camera **92***a* may be useful in detecting existence of an object, type of an object, identity of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In some aspects, Camera **92***a* may be or comprises a motion picture camera that can capture streams of pictures (i.e. motion pictures, videos, etc.). In other aspects, Camera **92***a* may be or comprises a still picture camera that can capture still pictures (i.e. photographs, etc.). In further aspects, Camera **92***a* may be or comprises a stereo camera (i.e. camera with multiple lenses, etc.) that can capture stereoscopic or range pictures. In further aspects, Camera **92***a* may be or comprises any other Camera **92***a*. In general, Camera **92***a* may capture any light (i.e. visible light, infrared light, ultraviolet light, x-ray light, etc.) across the electromagnetic spectrum onto a light-sensitive material. Any other technique known in art can be utilized to facilitate Camera **92***a* functionalities. In one example, a digital Camera **92***a* can utilize a charge coupled device (CCD), a complementary metal-oxide-semiconductor (CMOS) sensor, and/or other electronic image sensor to capture digital pictures that can then be stored in a memory or storage, or transmitted to any of the disclosed or other elements for further processing. In another example, analog Camera **92***a* can utilize an analog-to-digital converter to produce digital pictures. In some embodiments, Camera **92***a* can be built, embedded, or integrated in Device **98** and/or other disclosed element. In other embodiments, Camera **92***a* can be an external Camera **92***a* connected with Device **98** and/or other disclosed element. In further embodiments, Camera **92***a* comprises Computing Device **70** or elements thereof. In general, Camera **92***a* can be implemented in any suitable configuration to provide its functionalities. Camera **92***a* may capture one or more digital pictures. A digital picture may include a collection of color encoded pixels or dots. Examples of file formats that can be utilized to store a digital picture include JPEG, GIF, TIFF, PNG, PDF, and/or other digitally encoded picture formats. A stream of digital pictures (i.e. motion picture, video, etc.) may include one or more digital pictures. Examples of file formats that can be utilized to store a stream of digital pictures include MPEG, AVI, FLV, MOV, RM, SWF, WMV, DivX, and/or other digitally encoded motion picture formats.

In other embodiments, Sensor **92** may be or include Microphone **92***b* as shown in FIG. **3**B. Microphone **92***b* comprises the functionality for capturing one or more sounds, and/or other functionalities. As such, Microphone **92***b* can be used to capture sounds from Device's **98** surrounding. Microphone **92***b* may be useful in detecting existence of an object, type of an object, identity of an object, bearing/angle of an object, activity (i.e. motion, sounding, etc.) of an object, and/or other properties of an object. In some aspects, Microphone **92***b* may be omnidirectional microphone that enables capturing sounds from any direction. In other aspects, Microphone **92***b* may be a directional (i.e. unidirectional, bidirectional, etc.) microphone that enables capturing sounds from one or more directions while ignoring or being insensitive to sounds from other directions. In general, Microphone **92***b* may utilize a membrane sensitive to air pressure and may produce electrical signal from air pressure variations. Samples of the electrical signal can then be read to produce a stream of digital sound samples. Any other technique known in art can be utilized to facilitate Microphone **92***b* functionalities. In one example, a digital Microphone **92***b* may include an integrated analog-to-digital converter to capture a stream of digital sound samples that can then be stored in a memory or storage, or transmitted to any of the disclosed or other elements for further processing. In another example, analog Microphone **92***b* may utilize an external analog-to-digital converter to produce a stream of digital sound samples. In some embodiments, Microphone **92***b* can be built, embedded, or integrated in Device **98**. In other embodiments, Microphone **92***b* can be an external Microphone **92***b* connected with Device **98**. In further embodiments where used in water, Microphone **92***b* may be or include a hydrophone. In further embodiments, Microphone **92***b* comprises Computing Device **70** or elements thereof. In general, Microphone **92***b* can be implemented in any suitable configuration to provide its functionalities. Examples of file formats that can be utilized to store a stream of digital sound samples include WAV, WMA, AIFF, MP3, RA, OGG, and/or other digitally encoded sound formats.

In further embodiments, Sensor **92** may be or include Lidar **92***c* as shown in FIG. **3**C. Lidar **92***c* may be useful in

detecting existence of an object, type of an object, identity of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In some aspects, Lidar **92***c* may emit a light signal (i.e. laser beam, etc.) and listen for a signal that is reflected or backscattered from an object. Any other technique known in art can be utilized to facilitate Lidar **92***c* functionalities.

In further embodiments, Sensor **92** may be or include a Radar **92***d* as shown in FIG. **3**D. Radar **92***d* may be useful in detecting existence of an object, type of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In some aspects, Radar **92***d* may emit a radio signal (i.e. radio wave, etc.) and listen for a signal that is reflected or backscattered from an object. Any other technique known in art can be utilized to facilitate Radar **92***d* functionalities.

In further embodiments, Sensor **92** may be or include Sonar **92***e* as shown in FIG. **3**E. Sonar **92***e* may be useful in detecting existence of an object, type of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In some aspects, Sonar **92***e* may emit a sound signal (i.e. sound pulse, etc.) and listen for a signal that is reflected or backscattered from an object. Any other technique known in art can be utilized to facilitate Sonar **92***e* functionalities.

One of ordinary skill in art will understand that the aforementioned sensors are described merely as examples of a variety of possible implementations, and that while all possible sensors are too voluminous to describe, other sensors known in art that can facilitate detecting of objects and/or their properties in Device's **98** surrounding are within the scope of this disclosure. Any combination of the aforementioned and/or other sensors can be used in various embodiments.

Object Processing Unit **93** comprises the functionality for processing output from Sensor **92** to obtain information of interest, and/or other functionalities. As such, Object Processing Unit **93** can be used to process output from Sensor **92** to detect objects and/or their properties in Device's **98** surrounding. In some embodiments, Object Processing Unit **93** comprises the functionality for creating or generating Collection of Object Representations **525** (also referred to as Coll of Obj Rep or other suitable name or reference) and storing one or more Object Representations **625** (also referred to simply as object representations, representations of objects, or other suitable name or reference), Object Properties **630** (also referred to simply as object properties or other suitable name or reference), and/or other elements or information into the Collection of Object Representations **525**. As such, Collection of Object Representations **525** comprises the functionality for storing one or more Object Representations **625**, Object Properties **630**, and/or other elements or information. Object Representation **625** may include an electronic representation of an object (i.e. Object **615** [later described], etc.) detected in Device's **98** surrounding. In some aspects, Collection of Object Representations **525** includes one or more Object Representations **625**, Object Properties **630**, and/or other elements or information related to objects detected in Device's **98** surrounding at a particular time. Collection of Object Representations **525** may, therefore, include knowledge (i.e. unit of knowledge, etc.) of Device's **98** circumstances including objects with various properties at a particular time. In some designs, a Collection of Object Representations **525** may include or be associated with a time stamp (not shown), order (not shown), or other time related information. For example, one Collection of Object Representations **525** may be associated with time stamp t**1**, another Collection of Object Representations **525** may be associated with time stamp t**2**, and so on. Time stamps t**1**, t**2**, etc. may indicate the times of generating Collections of Object Representations **525**, for instance. In other embodiments, Object Processing Unit **93** comprises the functionality for creating or generating a stream of Collections of Object Representations **525**. A stream of Collections of Object Representations **525** may include one Collection of Object Representations **525** or a group, sequence, or other plurality of Collections of Object Representations **525**. In some aspects, a stream of Collections of Object Representations **525** includes one or more Collections of Object Representations **525**, and/or other elements or information related to objects detected in Device's **98** surrounding over time. A stream of Collections of Object Representations **525** may, therefore, include knowledge (i.e. unit of knowledge, etc.) of Device's **98** circumstances including objects with various properties over time. As circumstances including objects with various properties in Device's **98** surrounding change (i.e. objects and/or their properties change, move, act, transform, etc.) over time, this change may be captured in a stream of Collections of Object Representations **525**. In some designs, each Collection of Object Representations **525** in a stream may include or be associated with the aforementioned time stamp, order, or other time related information. For example, one Collection of Object Representations **525** in a stream may be associated with order **1**, a next Collection of Object Representations **525** in the stream may be associated with order **2**, and so on. Orders **1**, **2**, etc. may indicate the orders or places of Collections of Object Representations **525** within a stream (i.e. sequence, etc.), for instance. Examples of objects include biological objects (i.e. persons, animals, vegetation, etc.), nature objects (i.e. rocks, bodies of water, etc.), manmade objects (i.e. buildings, streets, ground/aerial/aquatic vehicles, etc.), and/or others. In some aspects, any part of an object may be detected as an object itself. For instance, instead of or in addition to detecting a vehicle as an object, a wheel and/or other parts of the vehicle may be detected as objects. In general, object may include any object or part thereof that can be detected. Examples of object properties include existence of an object, type of an object (i.e. person, cat, vehicle, building, street, tree, rock, etc.), identity of an object (i.e. name, identifier, etc.), distance of an object, bearing/angle of an object, location of an object (i.e. distance and bearing/angle from a known point, coordinates, etc.), shape/size of an object (i.e. height, width, depth, computer model, point cloud, etc.), activity of an object (i.e. motion, gestures, etc.), and/or other properties of an object. Type of an object, for example, may include any classification of objects ranging from detailed such as person, cat, vehicle, building, street, tree, rock, etc. to generalized such as biological object, nature object, manmade object, etc., and/or others including their sub-types. Location of an object, for example, can include a relative location such as one defined by distance and bearing/angle from a known point or location (i.e. Device **98** location, etc.). Location of an object, for example, can also include absolute location such as one defined by object coordinates. In general, an object property may include any attribute of an object (i.e. existence of an object, type of an object, identity of an object, shape/size of an object, etc.), any relationship of an object with Device **98**, other objects, or the environment (i.e. distance of an object, bearing/angle of an object, friend/foe relationship, etc.),

and/or other information related to an object. In some implementations, Object Processing Unit **93** and/or any of its elements or functionalities can be included in Sensor **92**. In other implementations, Object Processing Unit **93** and/or any of its elements or functionalities can be embedded into or operate on Processor **11**. In further implementations, Object Processing Unit **93** and/or any of its elements or functionalities can be embedded into or operate in DCADO Unit **100**, and/or other disclosed elements. Object Processing Unit **93** may be provided in any suitable configuration. Object Processing Unit **93** may include any signal processing techniques or elements known in art as applicable.

In some embodiments, Object Processing Unit **93** may include Picture Recognizer **94***a* as shown in FIG. **3**A. Picture Recognizer **94***a* comprises the functionality for detecting or recognizing objects and/or their properties in visual data, and/or other disclosed functionalities. Visual data includes digital motion pictures, digital still pictures, and/or other visual data. Examples of file formats that can be utilized to store visual data include AVI, DivX, MPEG, JPEG, GIF, TIFF, PNG, PDF, and/or other file formats. For example, Picture Recognizer **94***a* can be used for detecting or recognizing objects and/or their properties in one or more digital pictures captured by one or more Cameras **92***a*. Picture Recognizer **94***a* can be utilized in detecting or recognizing existence of an object, type of an object, identity of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In general, Picture Recognizer **94***a* can be used for any operation supported by Picture Recognizer **94***a*. Picture Recognizer **94***a* may detect or recognize an object and/or its properties as well as track the object and/or its properties in one or more digital pictures or streams of digital pictures (i.e. motion pictures, video, etc.). In the case of a person, Picture Recognizer **94***a* may detect or recognize a human head or face, upper body, full body, or portions/combinations thereof. In some aspects, Picture Recognizer **94***a* may detect or recognize objects and/or their properties from a digital picture by comparing regions of pixels from the digital picture with collections of pixels comprising known objects and/or their properties. The collections of pixels comprising known objects and/or their properties can be learned or manually, programmatically, or otherwise defined. The collections of pixels comprising known objects and/or their properties can be stored in any data structure or repository (i.e. one or more files, database, etc.) that resides locally on Device **98**, or remotely on a remote computing device (i.e. server, cloud, etc.) accessible over a network or an interface. In other aspects, Picture Recognizer **94***a* may detect or recognize objects and/or their properties from a digital picture by comparing features (i.e. lines, edges, ridges, corners, blobs, regions, etc.) of the digital picture with features of known objects and/or their properties. The features of known objects and/or their properties can be learned or manually, programmatically, or otherwise defined. The features of known objects and/or their properties can be stored in any data structure or repository (i.e. neural network, one or more files, database, etc.) that resides locally on Device **98**, or remotely on a remote computing device (i.e. server, cloud, etc.) accessible over a network or an interface. Typical steps or elements in a feature oriented picture recognition include pre-processing, feature extraction, detection/segmentation, decision-making, and/or others, or a combination thereof, each of which may include its own sub-steps or sub-elements depending on the application. In further aspects, Picture Recognizer **94***a* may detect or recognize multiple objects and/or their properties from a digital picture using the aforementioned pixel or feature comparisons, and/or other detection or recognition techniques. For example, a picture may depict two objects in two of its regions both of which Picture Recognizer **94***a* can detect simultaneously. In further aspects, where objects and/or their properties span multiple pictures, Picture Recognizer **94***a* may detect or recognize objects and/or their properties by applying the aforementioned pixel or feature comparisons and/or other detection or recognition techniques over a stream of digital pictures (i.e. motion picture, video, etc.). For example, once an object is detected in a digital picture (i.e. frame, etc.) of a stream of digital pictures (i.e. motion picture, video, etc.), the region of pixels comprising the detected object or the object's features can be searched in other pictures of the stream of digital pictures, thereby tracking the object through the stream of digital pictures. In further aspects, Picture Recognizer **94***a* may detect or recognize an object's activities by identifying and/or analyzing differences between a detected region of pixels of one picture (i.e. frame, etc.) and detected regions of pixels of other pictures in a stream of digital pictures. For example, a region of pixels comprising a person's face can be detected in multiple consecutive pictures of a stream of digital pictures (i.e. motion picture, video, etc.). Differences among the detected regions of the consecutive pictures may be identified in the mouth part of the person's face to indicate smiling or speaking activity. In further aspects, Picture Recognizer **94***a* may detect or recognize objects and/or their properties using one or more artificial neural networks, which may include statistical techniques. Examples of artificial neural networks that can be used in Picture Recognizer **94***a* include convolutional neural networks (CNNs), time delay neural networks (TDNNs), deep neural networks, and/or others. In one example, picture recognition techniques and/or tools involving convolutional neural networks may include identifying and/or analyzing tiled and/or overlapping regions or features of a digital picture, which may then be used to search for pictures with matching regions or features. In another example, features of different convolutional neural networks responsible for spatial and temporal streams can be fused to detect objects and/or their properties in streams of digital pictures (i.e. motion pictures, videos, etc.). In general, Picture Recognizer **94***a* may include any machine learning, deep learning, and/or other artificial intelligence techniques. In further aspects, Picture Recognizer **94***a* can detect distance of a recognized object in a picture captured by a camera using structured light, sheet of light, or other lighting schemes, and/or by using phase shift analysis, time of flight, interferometry, or other techniques. In further aspects, Picture Recognizer **94***a* may detect distance of a recognized object in a picture captured by a stereo camera by using triangulation and/or other techniques. In further aspects, Picture Recognizer **94***a* may detect bearing/angle of a recognized object relative to the camera-facing direction by measuring the distance from the vertical centerline of the picture to a pixel in the recognized object based on known picture resolution and camera's angle of view. Any other techniques known in art can be utilized in Picture Recognizer **94***a*. For example, thresholds for similarity, statistical techniques, and/or optimization techniques can be utilized to determine a match in any of the above-described detection or recognition techniques. In some exemplary embodiments, object recognition techniques and/or tools such as OpenCV (Open Source Computer Vision) library, CamFind API, Kooaba, 6px API, Dextro API, and/or others can be utilized for detecting or recognizing objects and/or their properties in

digital pictures. In some aspects, picture recognition techniques and/or tools involve identifying and/or analyzing features such as lines, edges, ridges, corners, blobs, regions, and/or their relative positions, sizes, shapes, etc., which may then be used to search for pictures with matching features. For example, OpenCV library can detect an object (i.e. person, animal, vehicle, rock, etc.) and/or its properties in one or more digital pictures captured by Camera **92***a* or stored in an electronic repository, which can then be utilized in DCADO Unit **100**, Artificial Intelligence Unit **110**, and/or other elements. In other exemplary embodiments, facial recognition techniques and/or tools such as OpenCV (Open Source Computer Vision) library, Animetrics FaceR API, Lambda Labs Facial Recognition API, Face++ SDK, Neven Vision (also known as N-Vision) Engine, and/or others can be utilized for detecting or recognizing faces in digital pictures. In some aspects, facial recognition techniques and/or tools involve identifying and/or analyzing facial features such as the relative position, size, and/or shape of the eyes, nose, cheekbones, jaw, etc., which may then be used to search for pictures with matching features. For example, FaceR API can detect a person's face in one or more digital pictures captured by Camera **92***a* or stored in an electronic repository, which can then be utilized in DCADO Unit **100**, Artificial Intelligence Unit **110**, and/or other elements.

Various aspects or properties of digital pictures or pixels can be taken into account by Picture Recognizer **94***a* in any of the recognizing or comparisons. Examples of such aspects or properties include color adjustment, size adjustment, content manipulation, transparency (i.e. alpha channel, etc.), use of mask, and/or others. In some implementations, as digital pictures can be captured by various picture taking equipment, in various environments, and under various lighting conditions, Picture Recognizer **94***a* can adjust lighting or color of pixels or otherwise manipulate pixels before or during comparison. Lighting or color adjustment (also referred to as gray balance, neutral balance, white balance, etc.) may generally include manipulating or rebalancing the intensities of the colors (i.e. red, green, and/or blue if RGB color model is used, etc.) of one or more pixels. For example, Picture Recognizer **94***a* can adjust lighting or color of some or all pixels of one picture to make it more comparable to another picture. Picture Recognizer **94***a* can also incrementally adjust the pixels such as increasing or decreasing the red, green, and/or blue pixel values by a certain amount in each cycle of comparisons in order to find a substantially similar match at one of the incremental adjustment levels. Any of the publically available, custom, or other lighting or color adjustment techniques or programs can be utilized such as color filters, color balancing, color correction, and/or others. In other implementations, Picture Recognizer **94***a* can resize or otherwise transform a digital picture before or during comparison. Such resizing or transformation may include increasing or decreasing the number of pixels of a digital picture. For example, Picture Recognizer **94***a* can increase or decrease the size of a digital picture proportionally (i.e. increase or decrease length and/or width keeping aspect ratio constant, etc.) to equate its size with the size of another digital picture. Picture Recognizer **94***a* can also incrementally resize a digital picture such as increasing or decreasing the size of the digital picture proportionally by a certain amount in each cycle of comparisons in order to find a substantially similar match at one of the incremental sizes. Any of the publically available, custom, or other digital picture resizing techniques or programs can be utilized such as nearest-neighbor interpolation,

bilinear interpolation, bicubic interpolation, and/or others. In further implementations, Picture Recognizer **94***a* can manipulate content (i.e. all pixels, one or more regions, one or more depicted objects, etc.) of a digital picture before or during comparison. Such content manipulation may include moving, centering, aligning, resizing, transforming, and/or otherwise manipulating content of a digital picture. For example, Picture Recognizer **94***a* can move, center, or align content of one picture to make it more comparable to another picture. Any of the publically available, custom, or other digital picture manipulation techniques or programs can be utilized such as pixel moving, warping, distorting, aforementioned interpolations, and/or others. In further implementations, in digital pictures comprising transparency features or functionalities, Picture Recognizer **94***a* can utilize a threshold for acceptable number or percentage transparency difference. Alternatively, transparency can be applied to one or more pixels of a digital picture and color difference may then be determined between compared pixels taking into account the transparency related color effect. Alternatively, transparent pixels can be excluded from comparison. In further implementations, certain regions or subsets of pixels can be ignored or excluded during comparison using a mask. In general, any region or subset of a picture determined to contain no content of interest can be excluded from comparison using a mask. Examples of such regions or subsets include background, transparent or partially transparent regions, regions comprising insignificant content, or any arbitrary region or subset. Picture Recognizer **94***a* can perform any other pre-processing or manipulation of digital pictures or pixels before or during recognizing or comparison.

In other embodiments, Object Processing Unit **93** may include Sound Recognizer **94***b* as shown in FIG. **3**B. Sound Recognizer **94***b* comprises the functionality for detecting or recognizing objects and/or their properties in audio data, and/or other disclosed functionalities. Audio data includes digital sound, and/or other audio data. Examples of file formats that can be utilized to store audio data include WAV, WMA, AIFF, MP3, RA, OGG, and/or other file formats. For example, Sound Recognizer **94***b* can be used for detecting or recognizing objects and/or their properties in a stream of digital sound samples captured by one or more Microphones **92***b*. In the case of a person, Sound Recognizer **94***b* may detect or recognize human voice. Sound Recognizer **94***b* can be utilized in detecting or recognizing existence of an object, type of an object, identity of an object, bearing/angle of an object, activity (i.e. motion, sounding, etc.) of an object, and/or other properties of an object. In general, Sound Recognizer **94***b* can be used for any operation supported by Sound Recognizer **94***b*. In some aspects, Sound Recognizer **94***b* may detect or recognize an object and/or its properties from a stream of digital sound samples by comparing collections of sound samples from the stream of digital sound samples with collections of sound samples of known objects and/or their properties. The collections of sound samples of known objects and/or their properties can be learned, or manually, programmatically, or otherwise defined. The collections of sound samples of known objects and/or their properties can be stored in any data structure or repository (i.e. one or more files, database, etc.) that resides locally on Device **98**, or remotely on a remote computing device (i.e. server, cloud, etc.) accessible over a network or an interface. In other aspects, Sound Recognizer **94***b* may detect or recognize an object and/or its properties from a stream of digital sound samples by comparing features from the stream of digital sound samples with features of sounds

of known objects and/or their properties. The features of sounds of known objects and/or their properties can be learned, or manually, programmatically, or otherwise defined. The features of sounds of known objects and/or their properties can be stored in any data structure or repository (i.e. one or more files, database, neural network, etc.) that resides locally on Device **98**, or remotely on a remote computing device (i.e. server, cloud, etc.) accessible over a network or an interface. Typical steps or elements in a feature oriented sound recognition include pre-processing, feature extraction, acoustic modeling, language modeling, and/or others, or a combination thereof, each of which may include its own sub-steps or sub-elements depending on the application. In further aspects, Sound Recognizer **94*b*** may detect or recognize a variety of sounds from a stream of digital sound samples using the aforementioned sound sample or feature comparisons, and/or other detection or recognition techniques. For example, sound of a person, animal, vehicle, and/or other sounds can be detected by Sound Recognizer **94*b***. In further aspects, Sound Recognizer **94*b*** may detect or recognize sounds using Hidden Markov Models (HMM), Artificial Neural Networks, Dynamic Time Warping (DTW), Gaussian Mixture Models (GMM), and/or other models or techniques, or a combination thereof. Some or all of these models or techniques may include statistical techniques. Examples of artificial neural networks that can be used in Sound Recognizer **94*b*** include recurrent neural networks, time delay neural networks (TDNNs), deep neural networks, convolutional neural networks, and/or others. In general, Sound Recognizer **94*b*** may include any machine learning, deep learning, and/or other artificial intelligence techniques. In further aspects, Sound Recognizer **94*b*** may detect bearing/angle of a recognized object by measuring the direction in which Microphone **92*b*** is pointing when sound of maximum strength is received, by analyzing amplitude of the sound, by performing phase analysis (i.e. with microphone array, etc.) of the sound, and/or by utilizing other techniques. Any other techniques known in art can be utilized in Sound Recognizer **94*b***. For example, thresholds for similarity, statistical techniques, and/or optimization techniques can be utilized to determine a match in any of the above-described detection or recognition techniques. In some exemplary embodiments, operating system's Sound recognition functionalities such as iOS's Voice Services, Siri, and/or others can be utilized in Sound Recognizer **94*b***. For example, iOS Voice Services can detect an object (i.e. person, etc.) and/or its properties in a stream of digital sound samples captured by Microphone **92*b*** or stored in an electronic repository, which can then be utilized in DCADO Unit **100**, Artificial Intelligence Unit **110**, and/or other elements. In other exemplary embodiments, Java Speech API (JSAPI) implementation such as The Cloud Garden, Sphinx, and/or others can be utilized in Sound Recognizer **94*b***. For example, Cloud Garden JSAPI can detect an object (i.e. person, animal, vehicle, etc.) and/or its properties in a stream of digital sound samples captured by Microphone **92*b*** or stored in an electronic repository, which can then be utilized in DCADO Unit **100**, Artificial Intelligence Unit **110**, and/or other elements. Any other programming language's or platform's speech or sound processing API can similarly be utilized. In further exemplary embodiments, applications or engines providing Sound recognition functionalities such as HTK (Hidden Markov Model Toolkit), Kaldi, OpenEars, Dragon Mobile, Julius, iSpeech, CeedVocal, and/or others can be utilized in Sound Recognizer **94*b***. For example, Kaldi SDK can detect an object (i.e. person, animal, vehicle, etc.) and/or its properties in a stream of digital sound samples captured by Microphone **92*b*** or stored in an electronic repository, which can then be utilized in DCADO Unit **100**, Artificial Intelligence Unit **110**, and/or other elements.

In further embodiments, Object Processing Unit **93** may include Lidar Processing Unit **94*c*** as shown in FIG. **3**C. Lidar Processing Unit **94*c*** comprises the functionality for detecting or recognizing objects and/or their properties using light, and/or other disclosed functionalities. As such, Lidar Processing Unit **94*c*** can be utilized in detecting existence of an object, type of an object, identity of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In general, Lidar Processing Unit **94*c*** can be used for any operation supported by Lidar Processing Unit **94*c***. In one example, Lidar Processing Unit **94*c*** may detect distance of an object by measuring time delay between emission of a light signal (i.e. laser beam, etc.) and return of the light signal reflected from the object based on known speed of light. In another example, Lidar Processing Unit **94*c*** may detect bearing/angle of an object by analyzing the amplitudes of a light signal received by an array of detectors (i.e. detectors arranged into a quadrant or other arrangement, etc.). In a further example, Lidar Processing Unit **94*c*** may detect existence, type, identity, shape/size, activity, and/or other properties of an object by illuminating the object with light and acquiring an image of the object, which can then be processed using some of the previously described or other picture recognition techniques. In a further example, Lidar Processing Unit **94*c*** may detect existence, type, identity, shape/size, activity, and/or other properties of an object by illuminating the object with light and acquiring a point cloud representation of the object. Lidar Processing Unit **94*c*** may detect objects and/or their properties by utilizing any lidar or light-related techniques known in art.

In further embodiments, Object Processing Unit **93** may include Radar Processing Unit **94*d*** as shown in FIG. **3**D. Radar Processing Unit **94*d*** comprises the functionality for detecting or recognizing objects and/or their properties using radio waves, and/or other disclosed functionalities. As such, Radar Processing Unit **94*d*** can be utilized in detecting existence of an object, type of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In general, Radar Processing Unit **94*d*** can be used for any operation supported by Radar Processing Unit **94*d***. In one example, Radar Processing Unit **94*d*** may detect existence of an object by emitting a radio signal and listening for the radio signal reflected from the object. In another example, Radar Processing Unit **94*d*** may detect distance of an object by measuring time delay between emission of a radio signal and return of the radio signal reflected from the object based on known speed of the radio signal. In a further example, Radar Processing Unit **94*d*** may detect bearing/angle of an object by measuring the direction in which the antenna is pointing when the return signal of maximum strength is received, by analyzing amplitude of the return signal, by performing phase analysis (i.e. with antenna array, etc.) of the return signal, and/or by utilizing any amplitude, phase, or other techniques. In a further example, Radar Processing Unit **94*d*** may detect existence, type, identity, shape/size, activity, and/or other properties of an object by illuminating the object with radio waves and acquiring an image of the object, which can then be processed using some of the previously described or other picture recognition techniques. Radar Processing Unit **94*d***

may detect objects and/or their properties by utilizing any radar or radio-related techniques known in art.

In further embodiments, Object Processing Unit **93** may include Sonar Processing Unit **94*e*** as shown in FIG. **3**E. Sonar Processing Unit **94*e*** comprises the functionality for detecting or recognizing objects and/or their properties using sound, and/or other disclosed functionalities. As such, Sonar Processing Unit **94*e*** can be utilized in detecting existence of an object, type of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In general, Sonar Processing Unit **94*e*** can be used for any operation supported by Sonar Processing Unit **94*e***. In one example, Sonar Processing Unit **94*e*** may detect existence of an object by emitting a sound signal and listening for the sound signal reflected from the object. In another example, Sonar Processing Unit **94*e*** may detect distance of an object by measuring time delay between emission of a sound signal and return of the sound signal reflected from the object based on known speed of the sound signal. In a further example, Sonar Processing Unit **94*e*** may detect bearing/angle of an object by measuring the direction in which the microphone is pointing when the return signal of maximum strength is received, by analyzing amplitude of the return signal, by performing phase analysis (i.e. with microphone array, etc.) of the return signal, and/or by utilizing any amplitude, phase, or other techniques. In a further example, Sonar Processing Unit **94*e*** may detect existence, type, identity, shape/size, activity, and/or other properties of an object by illuminating the object with sound pulses and acquiring an image of the object, which can then be processed using some of the previously described or other picture recognition techniques. Sonar Processing Unit **94*e*** may detect objects and/or their properties by utilizing any sonar or sound-related techniques known in art.

One of ordinary skill in art will understand that the aforementioned techniques for detecting or recognizing objects and/or their properties are described merely as examples of a variety of possible implementations, and that while all possible techniques for detecting or recognizing objects and/or their properties are too voluminous to describe, other techniques for detecting or recognizing objects and/or their properties known in art are within the scope of this disclosure. Also, any signal processing technique known in art that can facilitate the disclosed functionalities can be utilized in various embodiments. Any combination of the aforementioned and/or other sensors, object detecting or recognizing techniques, signal processing techniques, and/or other elements or techniques can be used in various embodiments.

Referring to FIGS. **4**A-**4**B, an exemplary embodiment of Objects **615** (also referred to simply as objects or other suitable name or reference) detected in Device's **98** surrounding, and resulting Collection of Object Representations **525** are illustrated.

As shown for example in FIG. **4**A, Object **615*a*** is detected. Object **615*a*** may be recognized as a cat. Object **615*a*** may be detected at a distance of 6 m from Device **98**. Object **615*a*** may be detected at a bearing/angle of 56° from Device's **98** centerline. Furthermore, Object **615*b*** is also detected. Object **615*b*** may be recognized as a tree. Object **615*b*** may be detected at a distance of 10 m from Device **98**. Object **615*b*** may be detected at a bearing/angle of **131** from Device's **98** centerline. Furthermore, Object **615*c*** is also detected. Object **615*c*** may be recognized as a person. Object **615*c*** may be identified as John Doe. Object **615*c*** may be detected at a distance of 8 m from Device **98**. Object **615*c*** may be detected at a bearing/angle of 287° from Device's **98** centerline. Any other Objects **615** instead of or in addition to Object **615*a***, Object **615*b***, and Object **615*c*** may be detected. In some aspects, any features, functionalities, and embodiments of Camera **92*a***/Picture Recognizer **94*a***, Microphone **92*b***/Sound Recognizer **94*b***, and/or other sensors or techniques can be utilized for recognizing and/or identifying a person, a cat, a tree, and/or other Objects **615**. In further aspects, any features, functionalities, and embodiments of Camera **92*a***/Picture Recognizer **94*a***, Microphone **92*b***/Sound Recognizer **94*b***, Lidar **92*c***/Lidar Processing Unit **94*c***, Radar **92*d***/Radar Processing Unit **94*d***, Sonar **92*e***/Sonar Processing Unit **94*e***, and/or other sensors or techniques can be utilized for detecting distance, bearing/angle, and/or other object properties.

As shown for example in FIG. **4**B, Object Processing Unit **93** may create or generate Collection of Object Representations **525** including Object Representation **625*a*** representing Object **615*a***, Object Representation **625*b*** representing Object **615*b***, Object Representation **625*c*** representing Object **615*c***, etc. For instance, Object Representation **625*a*** may include Object Property **630*aa*** "Cat" in Category **635*aa*** "Type", Object Property **630*ab*** "6 m" in Category **635*ab*** "Distance", Object Property **630*ac*** "56°" in Category **635*ac*** "Bearing", etc. Also, Object Representation **625*b*** may include Object Property **630*ba*** "Tree" in Category **635*ba*** "Type", Object Property **630*bb*** "10 m" in Category **635*bb*** "Distance", Object Property **630*bc*** "131°" in Category **635*bc*** "Bearing", etc. Also, Object Representation **625*c*** may include Object Property **630*ca*** "Person" in Category **635*ca*** "Type", Object Property **630*cb*** "John Doe" in Category **635*cb*** "Identity", Object Property **630*cc*** "8 m" in Category **635*cc*** "Distance", Object Property **630*cd*** "287°" in Category **635*cd*** "Bearing", etc. Any number of Object Representations **625**, and/or other elements or information can be included in Collection of Object Representations **525**. Any number of Object Properties **630** (also referred to simply as object properties or other suitable name or reference), and/or other elements or information can be included in an Object Representation **625**. In some aspects, a reference to Collection of Object Representations **525** comprises a reference to a collection of Object Properties **630** and/or other elements or information related to one or more Objects **615**. Other additional Object Representations **625**, Object Properties **630**, elements, and/or information can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate embodiments of Collection of Object Representations **525**.

Referring now to DCADO Unit **100**, DCADO Unit **100** comprises any hardware, programs, or a combination thereof. DCADO Unit **100** comprises the functionality for learning the operation of Device **98** in circumstances including objects with various properties. DCADO Unit **100** comprises the functionality for structuring and/or storing this knowledge in a knowledgebase (i.e. neural network, graph, sequences, other repository, etc.). DCADO Unit **100** comprises the functionality for enabling autonomous operation of Device **98** in circumstances including objects with various properties. DCADO Unit **100** comprises the functionality for interfacing with or attaching to Application Program **18**, Processor **11**, Logic Circuit **250** (later described), and/or other processing element. DCADO Unit **100** comprises the functionality for obtaining instruction sets, data, and/or other information used, implemented, and/or executed by Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element.

DCADO Unit **100** comprises the functionality for modifying instruction sets, data, and/or other information used, implemented, and/or executed by Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. DCADO Unit **100** comprises learning, anticipating, decision making, automation, and/or other functionalities disclosed herein. Statistical, artificial intelligence, machine learning, and/or other models or techniques are utilized to implement the disclosed devices, systems, and methods.

When DCADO Unit **100** functionalities are applied on Application Program **18**, Processor **11**, Logic Circuit **250** (later described), and/or other processing element of Device **98**, Device **98** may become autonomous. DCADO Unit **100** may take control from, share control with, and/or release control to Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element to implement autonomous operation of Device **98**. DCADO Unit **100** may take control from, share control with, and/or release control to Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element automatically or after prompting User **50** to allow it. In some aspects, Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element of an autonomous Device **98** may include or be provided with anticipatory (also referred to as alternate or other suitable name or reference) instructions or instruction sets that User **50** did not issue or cause to be executed. Such anticipatory instructions or instruction sets include instruction sets that User **50** may want or is likely to issue or cause to be executed. Anticipatory instructions or instruction sets can be generated by DCADO Unit **100** or elements thereof based on Device's **98** circumstances including objects with various properties. As such, Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element of an autonomous Device **98** may include or be provided with some or all original instructions or instruction sets and/or any anticipatory instructions or instruction sets generated by DCADO Unit **100**. Therefore, autonomous Device **98** operating may include executing some or all original instructions or instruction sets and/or any anticipatory instructions or instruction sets generated by DCADO Unit **100**. In one example, DCADO Unit **100** can overwrite or rewrite the original instructions or instruction sets of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element with DCADO Unit **100**-generated instructions or instruction sets. In another example, DCADO Unit **100** can insert or embed DCADO Unit **100**-generated instructions or instruction sets among the original instructions or instruction sets of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. In a further example, DCADO Unit **100** can branch, redirect, or jump to DCADO Unit **100**-generated instructions or instruction sets from the original instructions or instruction sets of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element.

In some embodiments, autonomous Device **98** operating comprises determining, by DCADO Unit **100**, a next instruction or instruction set to be executed based on Device's **98** circumstances including objects with various properties prior to the user issuing or causing to be executed the next instruction or instruction set. In yet other embodiments, autonomous application operating comprises determining, by DCADO Unit **100**, a next instruction or instruction set to be executed based on Device's **98** circumstances including objects with various properties prior to the system receiving the next instruction or instruction set.

In some embodiments, autonomous Device **98** operating includes a partially or fully autonomous operating. In an

example involving partially autonomous Device **98** operating, a user confirms DCADO Unit **100**-generated instructions or instruction sets prior to their execution. In an example involving fully autonomous application operating, DCADO Unit **100**-generated instructions or instruction sets are executed without user or other system confirmation (i.e. automatically, etc.).

In some embodiments, a combination of DCADO Unit **100** and other systems and/or techniques can be utilized to implement Device's **98** operation. In one example, DCADO Unit **100** may be a primary or preferred system for implementing Device's **98** operation. While operating autonomously under the control of DCADO Unit **100**, Device **98** may encounter a circumstance including objects with various properties that has not been encountered or learned before. In such situations, User **50** and/or non-DCADO system may take control of Device's **98** operation. DCADO Unit **100** may take control again when Device **98** encounters a previously learned circumstance including objects with various properties. Naturally, DCADO Unit **100** can learn Device's **98** operation in circumstances while User **50** and/or non-DCADO system is in control of Device **98**, thereby reducing or eliminating the need for future involvement of User **50** and/or non-DCADO system. In another example, User **50** and/or non-DCADO system may be a primary or preferred system for implementing Device's **98** operation. While operating under the control of User **50** and/or non-DCADO system, User **50** and/or non-DCADO system may release control to DCADO Unit **100** for any reason (i.e. User **50** gets tired or distracted, non-DCADO system gets stuck or cannot make a decision, etc.), at which point Device **98** can be controlled by DCADO Unit **100**. In some designs, DCADO Unit **100** may take control in certain special circumstances including objects with various properties where DCADO Unit **100** may offer superior performance even though User **50** and/or non-DCADO system may generally be preferred. Once Device **98** leaves such special circumstances, DCADO Unit **100** may release control to User **50** and/or non-DCADO system. In general, DCADO Unit **100** can take control from, share control with, or release control to User **50**, non-DCADO system, and/or other system or process at any time, in any circumstances, and remain in control for any period of time as needed.

In some embodiments, DCADO Unit **100** may control one or more sub-devices, sub-systems, or elements of Device **98** while User **50** and/or non-DCADO system may control other one or more sub-devices, sub-systems, or elements of Device **98**.

It should be understood that a reference to autonomous operating of Device **98** may include autonomous operating of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element depending on context.

Referring now to Acquisition Interface **120**, Acquisition Interface **120** comprises the functionality for obtaining and/or receiving instruction sets, data, and/or other information. Acquisition Interface **120** comprises the functionality for obtaining and/or receiving instruction sets, data, and/or other information from Processor **11**, Application Program **18**, Logic Circuit **250** (later described), and/or other processing element. Acquisition Interface **120** comprises the functionality for obtaining and/or receiving instruction sets, data, and/or other information at runtime. In some aspects, an instruction set may include any computer command, instruction, signal, or input used in Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. Therefore, the terms instruction set, command, instruction, signal, input, or other such terms may be used

interchangeably herein depending on context. Acquisition Interface **120** also comprises the functionality for attaching to or interfacing with Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. In one example, Acquisition Interface **120** comprises the functionality to access and/or read runtime engine/environment, virtual machine, operating system, compiler, just-in-time (JIT) compiler, interpreter, translator, execution stack, file, object, data structure, and/or other computing system elements. In another example, Acquisition Interface **120** comprises the functionality to access and/or read memory, storage, bus, interfaces, and/or other computing system elements. In a further example, Acquisition Interface **120** comprises the functionality to access and/or read Processor **11** registers and/or other Processor **11** elements. In a further example, Acquisition Interface **120** comprises the functionality to access and/or read inputs and/or outputs of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. In a further example, Acquisition Interface **120** comprises the functionality to access and/or read functions, methods, procedures, routines, subroutines, and/or other elements of Application Program **18**. In a further example, Acquisition Interface **120** comprises the functionality to access and/or read source code, bytecode, compiled, interpreted, or otherwise translated code, machine code, and/or other code. In a further example, Acquisition Interface **120** comprises the functionality to access and/or read values, variables, parameters, and/or other data or information. Acquisition Interface **120** also comprises the functionality for transmitting the obtained instruction sets, data, and/or other information to Artificial Intelligence Unit **110** and/or other element. As such, Acquisition Interface **120** provides input into Artificial Intelligence Unit **110** for knowledge structuring, anticipating, decision making, and/or other functionalities later in the process. Acquisition Interface **120** also comprises other disclosed functionalities.

Acquisition Interface **120** can employ various techniques for obtaining instruction sets, data, and/or other information. In one example, Acquisition Interface **120** can attach to and/or obtain Processor's **11**, Application Program's **18**, Logic Circuits **250**, and/or other processing element's instruction sets, data, and/or other information through tracing or profiling techniques. Tracing or profiling may be used for outputting Processor's **11**, Application Program's **18**, Logic Circuits **250**, and/or other processing element's instruction sets, data, and/or other information at runtime. For instance, tracing or profiling may include adding trace code (i.e. instrumentation, etc.) to an application and/or outputting trace information to a specific target. The outputted trace information (i.e. instruction sets, data, and/or other information, etc.) can then be provided to or recorded into a file, data structure, repository, an application, and/or other system or target that may receive such trace information. As such, Acquisition Interface **120** can utilize tracing or profiling to obtain instruction sets, data, and/or other information and provide them as input into Artificial Intelligence Unit **110**. In some aspects, instrumentation can be performed in source code, bytecode, compiled, interpreted, or otherwise translated code, machine code, and/or other code. In other aspects, instrumentation can be performed in various elements of a computing system such as memory, virtual machine, runtime engine/environment, operating system, compiler, interpreter, translator, processor registers, execution stack, program counter, and/or other elements. In yet other aspects, instrumentation can be performed in various abstraction layers of a computing system such as in software layer (i.e. Application Program **18**, etc.), in virtual machine (if VM is used), in operating system, in Processor **11**, and/or in other layers or areas that may exist in a particular computing system implementation. In yet other aspects, instrumentation can be performed at various time periods in an application's execution such as source code write time, compile time, interpretation time, translation time, linking time, loading time, runtime, and/or other time periods. In yet other aspects, instrumentation can be performed at various granularities or code segments such as some or all lines of code, some or all statements, some or all instructions or instruction sets, some or all basic blocks, some or all functions/routines/subroutines, and/or some or all other code segments.

In some embodiments, Application Program **18** can be automatically instrumented. For example, Acquisition Interface **120** can access Application Program's **18** source code, bytecode, or machine code and select instrumentation points of interest. Selecting instrumentation points may include finding locations in the source code, bytecode, or machine code corresponding to function calls, function entries, function exits, object creations, object destructions, event handler calls, new lines (i.e. to instrument all lines of code, etc.), thread creations, throws, and/or other points of interest. Instrumentation code can then be inserted at the instrumentation points of interest to output Application Program's **18** instruction sets, data, and/or other information. In response to executing instrumentation code, Application Program's **18** instruction sets, data, and/or other information may be received by Acquisition Interface **120**. In some aspects, Application Program's **18** source code, bytecode, or machine code can be dynamically instrumented. For example, instrumentation code can be dynamically inserted into Application Program **18** at runtime.

In other embodiments, Application Program **18** can be manually instrumented. In one example, a programmer can instrument a function call by placing an instrumenting instruction immediately after the function call as in the following example.

```
Device1.moveForward(12);
traceApplication('Device1.moveForward(12);');
```

In another example, an instrumenting instruction can be placed immediately before the function call, or at the beginning, end, or anywhere within the function itself. A programmer may instrument all function calls or only function calls of interest. In a further example, a programmer can instrument all lines of code or only code lines of interest. In a further example, a programmer can instrument other elements utilized or implemented within Application Program **18** such as objects and/or any of their functions, data structures and/or any of their functions, event handlers and/or any of their functions, threads and/or any of their functions, and/or other elements or functions. Similar instrumentation as in the preceding examples can be performed automatically or dynamically. In some designs where manual code instrumentation is utilized, Acquisition Interface **120** can optionally be omitted and Application Program's **18** instruction sets, data, and/or other information may be transmitted directly to Artificial Intelligence Unit **110**.

In some embodiments, DCADO Unit **100** can be selective in learning instruction sets, data, and/or other information to those implemented, utilized, or related to an object, data structure, repository, thread, function, and/or other element of Application Program **18**. In some aspects, Acquisition Interface **120** can obtain Application Program's **18** instruc-

tion sets, data, and/or other information implemented, utilized, or related to a certain object in an object oriented Application Program **18**.

In some embodiments, various computing systems and/or platforms may provide native tools for obtaining instruction sets, data, and/or other information. Also, independent vendors may provide portable tools with similar functionalities that can be utilized across different computing systems and/or platforms. These native and portable tools may provide a wide range of functionalities to obtain runtime and other information such as instrumentation, tracing or profiling, logging application or system messages, outputting custom text messages, outputting objects or data structures, outputting functions/routines/subroutines or their invocations, outputting variable or parameter values, outputting thread or process behaviors, outputting call or other stacks, outputting processor registers, providing runtime memory access, providing inputs and/or outputs, performing live application monitoring, and/or other capabilities. One of ordinary skill in art will understand that, while all possible variations of the techniques to obtain instruction sets, data, and/or other information are too voluminous to describe, these techniques are within the scope of this disclosure.

In one example, obtaining instruction sets, data, and/or other information can be implemented through the .NET platform's native tools for application tracing or profiling such as System.Diagnostics.Trace, System.Diagnostics.Debug, and System.Diagnostics.TraceSource classes for tracing execution flow, and System.Diagnostics.Process, System.Diagnostics.EventLog, and System.Diagnostics.PerformanceCounter classes for profiling code, accessing local and remote processes, starting and stopping system processes, and interacting with Windows event logs, etc. For instance, a set of trace switches can be created that output an application's information. The switches can be configured using the .config file. For a Web application, this may typically be Web.config file associated with the project. In a Windows application, this file may typically be named applicationName.exe.config. Trace code can be added to application code automatically or manually as previously described. Appropriate listener can be created where the trace output is received. Trace code may output trace messages to a specific target such as a file, a log, a database, an object, a data structure, and/or other repository or system. Acquisition Interface **120** or Artificial Intelligence Unit **110** can then read or obtain the trace information from these targets. In some aspects, trace code may output trace messages directly to Acquisition Interface **120**. In other aspects, trace code may output trace messages directly to Artificial Intelligence Unit **110**. In the case of outputting trace messages to Acquisition Interface **120** or directly to Artificial Intelligence Unit **110**, custom listeners can be built to accommodate these specific targets. Other platforms, tools, and/or techniques can provide equivalent or similar functionalities as the above described ones.

In another example, obtaining instruction sets, data, and/or other information can be implemented through the .NET platform's Profiling API that can be used to create a custom profiler application for tracing, monitoring, interfacing with, and/or managing a profiled application. The Profiling API provides an interface that includes methods to notify the profiler of events in the profiled application. The Profiling API may also provide an interface to enable the profiler to call back into the profiled application to obtain information about the state of the profiled application. The Profiling API may further provide call stack profiling functionalities. Call stack (also referred to as execution stack, control stack, runtime stack, machine stack, the stack, etc.) includes a data structure that can store information about active subroutines of an application. The Profiling API may provide a stack snapshot method, which enables a trace of the stack at a particular point in time. The Profiling API may also provide a shadow stack method, which tracks the call stack at every instant. A shadow stack can obtain function arguments, return values, and information about generic instantiations. A function such as FunctionEnter can be utilized to notify the profiler that control is being passed to a function and can provide information about the stack frame and function arguments. A function such as FunctionLeave can be utilized to notify the profiler that a function is about to return to the caller and can provide information about the stack frame and function return value. An alternative to call stack profiling includes call stack sampling in which the profiler can periodically examine the stack. In some aspects, the Profiling API enables the profiler to change the in-memory code stream for a routine before it is just-in-time (JIT) compiled where the profiler can dynamically add instrumentation code to all or particular routines of interest. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through Java platform's APIs for application tracing or profiling such as Java Virtual Machine Profiling Interface (JVMPI), Java Virtual Machine Tool Interface (JVMTI), and/or other APIs or tools. These APIs can be used for instrumentation of an application, for notification of Java Virtual Machine (VM) events, and/or other functionalities. One of the tracing or profiling techniques that can be utilized includes bytecode instrumentation. The profiler can insert bytecodes into all or some of the classes. In application execution profiling, for example, these bytecodes may include methodEntry and methodExit calls. In memory profiling, for example, the bytecodes may be inserted after each new or after each constructor. In some aspects, insertion of instrumentation bytecode can be performed either by a post-compiler or a custom class loader. An alternative to bytecode instrumentation includes monitoring events generated by the JVMPI or JVMTI interfaces. Both APIs can generate events for method entry/exit, object allocation, and/or other events. In some aspects, JVMTI can be utilized for dynamic bytecode instrumentation where insertion of instrumentation bytecodes is performed at runtime. The profiler may insert the necessary instrumentation when a selected class is invoked in an application. This can be accomplished using the JVMTI's redefineClasses method, for example. This approach also enables changing of the level of profiling as the application is running. If needed, these changes can be made adaptively without restarting the application. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through JVMTI's programming interface that enables creation of software agents that can monitor and control a Java application. An agent may use the functionality of the interface to register for notification of events as they occur in the application, and to query and control the application. A JVMTI agent may use JVMTI functions to extract information from a Java application. A JVMTI agent can be utilized to obtain an application's runtime information such as method calls, memory allocation, CPU utilization, lock contention, and/or other information. JVMTI may include functions to obtain information about variables, fields, methods, classes, and/or

other information. JVMTI may also provide notification for numerous events such as method entry and exit, exception, field access and modification, thread start and end, and/or other events. Examples of JVMTI built-in methods include GetMethodName to obtain the name of an invoked method, GetThreadInfo to obtain information for a specific thread, GetClassSignature to obtain information about the class of an object, GetStackTrace to obtain information about the stack including information about stack frames, and/or other methods. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through java.lang.Runtime class that provides an interface for application tracing or profiling. Examples of methods provided in java.lang.Runtime that can be used to obtain an application's instruction sets, data, and/or other information include tracemethodcalls, traceinstructions, and/or other methods. These methods prompt the Java Virtual Machine to output trace information for a method or instruction in the virtual machine as it is executed. The destination of trace output may be system dependent and include a file, a listener, and/or other destinations where Acquisition Interface **120**, Artificial Intelligence Unit **110**, and/or other disclosed elements can access needed information. In addition to tracing or profiling tools native to their respective computing systems and/or platforms, many independent tools exist that provide tracing or profiling functionalities on more than one computing system and/or platform. Examples of these tools include Pin, DynamoRIO, KernInst, DynInst, Kprobes, OpenPAT, DTrace, SystemTap, and/or others. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through logging tools of the platform and/or operating system on which an application runs. Some logging tools may include nearly full feature sets of the tracing or profiling tools previously described. In one example, Visual Basic enables logging of runtime messages through its Microsoft.VisualBasic.Logging namespace that provides a log listener where the log listener may direct logging output to a file and/or other target. In another example, Java enables logging through its java.util.logging class. In some aspects, obtaining an application's instruction sets, data, and/or other information can be implemented through logging capabilities of the operating system on which an application runs. For example, Windows NT features centralized log service that applications and operating-system components can utilize to report their events including any messages. Windows NT provides functionalities for system, application, security, and/or other logging. An application log may include events logged by applications. Windows NT, for example, may include support for defining an event source (i.e. application that created the event, etc.). Windows Vista, for example, supports a structured XML log-format and designated log types to allow applications to more precisely log events and to help interpret the events. Examples of different types of event logs include administrative, operational, analytic, debug, and/or other log types including any of their subcategories. Examples of event attributes that can be utilized include eventID, level, task, opcode, keywords, and/or other event attributes. Windows wevtutil tool enables access to events, their structures, registered event publishers, and/or their configuration even before the events are fired. Wevtutil supports capabilities such as retrieval of the names of all logs on a computing device; retrieval of configuration information for a specific log; retrieval of event publishers on a computing device; reading events from an event log, from a log file, or using a structured query; exporting events from an event log, from a log file, or using a structured query to a specific target; and/or other capabilities. Operating system logs can be utilized solely if they contain sufficient information on an application's instruction sets, data, and/or other information. Alternatively, operating system logs can be utilized in combination with another source of information (i.e. trace information, call stack, processor registers, memory, etc.) to reconstruct the application's instruction sets, data, and/or other information needed for Artificial Intelligence Unit **110** and/or other elements. In addition to logging capabilities native to their respective platforms and/or operating systems, many independent tools exist that provide logging on different platforms and/or operating systems. Examples of these tools include Log 4j, Logback, SmartInspect, N Log, log 4net, Microsoft Enterprise Library, ObjectGuy Framework, and/or others. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through tracing or profiling the operating system on which an application runs. As in tracing or profiling an application, one of the techniques that can be utilized includes adding instrumentation code to the operating system's source code. Such instrumentation code can be added to the operating system's source code before kernel compilation or recompilation, for instance. This type of instrumentation may involve defining or finding locations in the operating system's source code where instrumentation code may be inserted. Kernel instrumentation can also be performed without the need for kernel recompilation or rebooting. In some aspects, instrumentation code can be added at locations of interest through binary rewriting of compiled kernel code. In other aspects, kernel instrumentation can be performed dynamically where instrumentation code is added and/or removed where needed at runtime. For instance, dynamic instrumentation may overwrite kernel code with a branch instruction that redirects execution to instrumentation code or instrumentation routine. In yet other aspects, kernel instrumentation can be performed using just-in-time (JIT) dynamic instrumentation where execution may be redirected to a copy of kernel's code segment that includes instrumentation code. This type of instrumentation may include a JIT compiler and creation of a copy of the original code segment having instrumentation code or calls to instrumentation routines embedded into the original code segment. Instrumentation of the operating system may enable total system visibility including visibility into an application's behavior by enabling generation of low level trace information. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through tracing or profiling the processor on which an application runs. For example, some Intel processors provide Intel Processor Trace (i.e. Intel PT, etc.), a low-level tracing feature that enables recording executed instruction sets, and/or other data or information of one or more applications. Intel PT is facilitated by the Processor Trace Decoder Library along with its related tools. Intel PT is a low-overhead execution tracing feature that records information about application execution on each hardware thread using dedicated hardware facilities. The recorded execution/trace information is

collected in data packets that can be buffered internally before being sent to a memory subsystem or another system or element (i.e. Acquisition Interface **120**, Artificial Intelligence Unit **110**, etc.). Intel PT also enables navigating the recorded execution/trace information via reverse stepping commands. Intel PT can be included in an operating system's core files and provided as a feature of the operating system. Intel PT can trace globally some or all applications running on an operating system. Acquisition Interface **120** or Artificial Intelligence Unit **110** can read or obtain the recorded execution/trace information from Intel PT. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through branch tracing or profiling. Branch tracing may include an abbreviated instruction trace in which only the successful branch instruction sets are traced or recorded. Branch tracing can be implemented through utilizing dedicated processor commands, for example. Executed branches may be saved into special branch trace store area of memory. With the availability and reference to a compiler listing of the application together with branch trace information, a full path of executed instruction sets can be reconstructed. The full path can also be reconstructed with a memory dump (containing the program storage) and branch trace information. In some aspects, branch tracing can be utilized for pre-learning or automated learning of an application's instruction sets, data, and/or other information where a number of application simulations (i.e. simulations of likely/common operations, etc.) are performed. As such, the application's operation can be learned automatically saving the time that would be needed to learn the application's operation directed by a user. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through assembly language. Assembly language is a low-level programming language for a computer or other programmable device in which there is a strong correlation between the language and the architecture's machine instruction sets. Syntax, addressing modes, operands, and/or other elements of an assembly language instruction set may translate directly into numeric (i.e. binary, etc.) representations of that particular instruction set. Because of this direct relationship with the architecture's machine instruction sets, assembly language can be a powerful tool for tracing or profiling an application's execution in processor registers, memory, and/or other computing system components. For example, using assembly language, memory locations of a loaded application can be accessed, instrumented, and/or otherwise manipulated. In some aspects, assembly language can be used to rewrite or overwrite original in-memory instruction sets of an application with instrumentation instruction sets. In other aspects, assembly language can be used to redirect application's execution to instrumentation routine/subroutine or other code segment elsewhere in memory by inserting a jump into the application's in-memory code, by redirecting program counter, or by other techniques. Some operating systems may implement protection from changes to applications loaded into memory. Operating system, processor, or other low level commands such as Linux mprotect command or similar commands in other operating systems may be used to unprotect the protected locations in memory before the change. In yet other aspects, assembly language can be used to obtain instruction sets, data, and/or other information through accessing and/or reading instruction register, program counter, other processor registers, memory locations, and/or other components of a computing system. In yet other aspects, high-level programming languages may call or execute an external assembly language program to facilitate obtaining instruction sets, data, and/or other information as previously described. In yet other aspects, relatively low-level programming languages such as C may allow embedding assembly language directly in their source code such as, for example, using asm keyword of C. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, it may be sufficient to obtain user or other inputs, variables, parameters, and/or other data in some procedural, simple object oriented, or other applications. In one instance, a simple procedural application executes a sequence of instruction sets until the end of the program. During its execution, the application may receive user or other input, store the input in a variable, and perform calculations using the variable to reach a result. The value of the variable can be obtained or traced. In another instance, a more complex procedural application comprises one or more functions/routines/subroutines each of which may include a sequence of instruction sets. The application may execute a main sequence of instruction sets with a branch to a function/routine/subroutine. During its execution, the application may receive user or other input, store the input in a variable, and pass the variable as a parameter to the function/routine/subroutine. The function/routine/subroutine may perform calculations using the parameter and return a value that the rest of the application can use to reach a result. The value of the variable or parameter passed to the function/routine/subroutine, and/or return value can be obtained or traced. Values of user or other inputs, variables, parameters, and/or other items of interest can be obtained through previously described tracing, instrumentation, and/or other techniques. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

Referring to FIG. **5**, in yet another example, obtaining instruction sets, data, and/or other information may be implemented through tracing, profiling, or sampling of instruction sets or data in processor registers, memory, or other computing system components where instruction sets, data, and/or other information may be stored or utilized. For example, Instruction Register **212** may be part of Processor **11** and it may store the instruction set currently being executed or decoded. In some processors, Program Counter **211** (also referred to as instruction pointer, instruction address register, instruction counter, or part of instruction sequencer) may be incremented after fetching an instruction set, and it may hold or point to the memory address of the next instruction set to be executed. In a processor where the incrementation precedes the fetch, Program Counter **211** may point to the current instruction set being executed. In the instruction cycle, an instruction set may be loaded into Instruction Register **212** after Processor **11** fetches it from location in Memory **12** pointed to by Program Counter **211**. Instruction Register **212** may hold the instruction set while it is decoded by Instruction Decoder **213**, prepared, and executed. In some aspects, data (i.e. operands, etc.) needed for instruction set execution may be loaded from Memory **12** into a register within Register Array **214**. In other aspects, the data may be loaded directly into Arithmetic Logic Unit **215**. For instance, as instruction sets pass through Instruction Register **212** during application execution, they may be transmitted to Acquisition Interface **120** as shown. Examples

of the steps in execution of a machine instruction set may include decoding the opcode (i.e. portion of a machine instruction set that may specify the operation to be performed), determining where the operands may be located (depending on architecture, operands may be in registers, the stack, memory, I/O ports, etc.), retrieving the operands, allocating processor resources to execute the instruction set (needed in some types of processors), performing the operation indicated by the instruction set, saving the results of execution, and/or other execution steps. Examples of the types of machine instruction sets that can be utilized include arithmetic, data handling, logical, program control, as well as special and/or other instruction set types. In addition to the ones described or shown, examples of other computing system or processor components that can be used during an instruction cycle include memory address register (MAR) that may hold the address of a memory block to be read from or written to; memory data register (MDR) that may hold data fetched from memory or data waiting to be stored in memory; data registers that may hold numeric values, characters, small bit arrays, or other data; address registers that may hold addresses used by instruction sets that indirectly access memory; general purpose registers (GPRs) that may store both data and addresses; conditional registers that may hold truth values often used to determine whether some instruction set should or should not be executed; floating point registers (FPRs) that may store floating point numbers; constant registers that may hold read-only values such as zero, one, or pi; special purpose registers (SPRs) such as status register, program counter, or stack pointer that may hold information on program state; machine-specific registers that may store data and settings related to a particular processor; Register Array **214** that may include an array of any number of processor registers; Arithmetic Logic Unit **215** that may perform arithmetic and logic operations; control unit that may direct processor's operation; and/or other circuits or components. Tracing, profiling, or sampling of processor registers, memory, or other computing system components can be implemented in a program, combination of hardware and program, or purely hardware system. Dedicated hardware may be built to perform tracing, profiling, or sampling of processor registers or any computing system components with marginal or no impact to computing overhead.

One of ordinary skill in art will recognize that FIG. **5** depicts one of many implementations of processor or computing system components, and that various additional components can be included, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate implementations. Processor or computing system components may be arranged or connected differently in alternate implementations. Processor or computing system components may also be connected with external elements using various connections. For instance, the connection between Instruction Register **212** and Acquisition Interface **120** may include any number or types of connections such as, for example, a dedicated connection for each bit of Instruction Register **212** (i.e. 32 connections for a 32 bit Instruction Register **212**, etc.). Any of the described or other connections or interfaces may be implemented among any processor or computing system components and Acquisition Interface **120** or other elements.

Referring to FIGS. **6**A-**6**B, in yet another example, obtaining instruction sets, data, and/or other information may be implemented through tracing, profiling, or sampling of Logic Circuit **250**. While Processor **11** includes any type or embodiment of logic circuit, Logic Circuit **250** is described separately here to offer additional detail on its functioning. Some Devices **98** may not need the processing capabilities of an entire Processor **11**, but instead a more tailored Logic Circuit **250**. Examples of such Devices **98** include home appliances, audio or video electronics, vehicle systems, toys, industrial machines, robots, and/or others. Logic Circuit **250** comprises the functionality for performing logic operations. Logic Circuit **250** comprises the functionality for performing logic operations using the circuit's inputs and producing outputs based on the logic operations performed on the inputs. Logic Circuit **250** may generally be implemented using transistors, diodes, and/or other electronic switches, but can also be constructed using vacuum tubes, electromagnetic relays (relay logic), fluidic logic, pneumatic logic, optics, molecules, or even mechanical elements. In some aspects, Logic Circuit **250** may be or include a microcontroller, field-programmable gate array (FPGA), application-specific integrated circuit (ASIC), and/or other computing circuit or device. In other aspects, Logic Circuit **250** may be or include any circuit or device comprising one or more logic gates, one or more transistors, one or more switches, and/or one or more other logic components. In further aspects, Logic Circuit **250** may be or include any integrated or other circuit or device that can perform logic operations. Logic may generally refer to Boolean logic utilized in binary operations, but other logics can also be used. Input into Logic Circuit **250** may include or refer to a value inputted into the Logic Circuit **250**, therefore, these terms may be used interchangeably herein depending on context. In one example, Logic Circuit **250** may perform some logic operations using four input values and produce two output values. As the four input values are delivered to or received by Logic Circuit **250**, they may be obtained by Acquisition Interface **120** through the four hardwired connections as shown in FIG. **6**A. In another example, Logic Circuit **250** may perform some logic operations using four input values and produce two output values. As the two output values are generated by or transmitted out of Logic Circuit **250**, they may be obtained by Acquisition Interface **120** through the two hardwired connections as shown in FIG. **6**B. In a further example, instead of or in addition to obtaining input and/or output values of Logic Circuit **250**, the state of Logic Circuit **250** may be obtained by reading or accessing values from one or more Logic Circuit's **250** internal components such as registers, memories, buses, and/or others (i.e. similar to the previously described tracing, profiling, and/or sampling of Processor **11** components, etc.). Tracing, profiling, or sampling of Logic Circuit **250** can be implemented in a program, combination of hardware and program, or purely hardware system. Dedicated hardware may be built to perform tracing, profiling, or sampling of Logic Circuit **250** with marginal or no impact to computing overhead. Any of the elements and/or techniques for tracing, profiling, or sampling of Logic Circuit **250** can similarly be implemented with Processor **11** and/or other processing elements. In some designs, DCADO Unit **100** may include clamps and/or other elements to attach DCADO Unit **100** to inputs (i.e. input wires, etc.) into and/or outputs (i.e. output wires, etc.) from Logic Circuit **250**. Such clamps and/or attachment elements enable seamless attachment of DCADO Unit **100** to any circuit or computing device without the need to redesign or alter the circuit or computing device.

In some embodiments, DCADO Unit **100** may learn input values directly from an actuator (previously described, not shown). For example, Logic Circuit **250** or other processing element may control an actuator that enables Device **98** to

perform mechanical, physical, and/or other operations. An actuator may receive one or more input values or control signals from Logic Circuit **250** or other processing element directing the actuator to perform specific operations. As one or more input values or control signals are delivered to or received by the actuator, they may be obtained by Acquisition Interface **120** as previously described with respect to obtaining input values of Logic Circuit **250**. Specifically, for instance, one or more input values or control signals of an actuator may be obtained by Acquisition Interface **120** via hardwired or other connections.

One of ordinary skill in art will recognize that FIGS. **6**A-**6**B depict one of many implementations of Logic Circuit **250** and that any number of input and/or output values can be utilized in alternate implementations. One of ordinary skill in art will also recognize that Logic Circuit **250** may include any number and/or combination of logic components to implement any logic operations.

Other additional techniques or elements may be utilized as needed for obtaining instruction sets, data, and/or other information, or some of the disclosed techniques or elements may be excluded, or a combination thereof may be utilized in alternate embodiments.

Referring to FIGS. **7**A-**7**E, some embodiments of Instruction Sets **526** are illustrated. In some aspects, Instruction Set **526** includes one or more instructions or commands of Application Program **18**. For example, Instruction Set **526** may include one or more instructions or commands of a high-level programming language such as Java or SQL, a low-level language such as assembly or machine language, an intermediate language or construct such as bytecode, and/or any other language or construct. In other aspects, Instruction Set **526** includes one or more inputs into and/or outputs from Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element. In further aspects, Instruction Set **526** includes one or more values or states of registers and/or other components of Logic Circuit **250**, Processor **11**, and/or other processing element. In general, Instruction Set **526** may include one or more instructions, commands, keywords, symbols (i.e. parentheses, brackets, commas, semicolons, etc.), operators (i.e. =, <, >, etc.), variables, values, objects, data structures, functions (i.e. Function1( ), FIRST( ), MIN( ), SQRT( ), etc.), parameters, states, signals, inputs, outputs, characters, digits, references thereto, and/or other components for performing an operation.

In an embodiment shown in FIG. **7**A, Instruction Set **526** includes code of a high-level programming language (i.e. Java, C++, etc.) comprising the following function call construct: Function1 (Parameter1, Parameter2, Parameter3, . . .). An example of a function cal applying the above construct includes the following Instruction Set **526**: moveTo(Device1, 14, 8). The function or reference thereto "moveTo(Device1, 14, 8)" may be an Instruction Set **526** directing Device1 to move to a location with coordinates 14 and 8, for example. In another embodiment shown in FIG. **7**B, Instruction Set **526** includes structured query language (SQL). In a further embodiment shown in FIG. **7**C, Instruction Set **526** includes bytecode (i.e. Java bytecode, Python bytecode, CLR bytecode, etc.). In a further embodiment shown in FIG. **7**D, Instruction Set **526** includes assembly code. In a further embodiment shown in FIG. **7**E, Instruction Set **526** includes machine code.

Referring to FIGS. **8**A-**8**B, some embodiments of Extra Information **527** (also referred to as Extra Info **527**) are illustrated. In an embodiment shown in FIG. **8**A, Collection of Object Representations **525** may include or be associated with Extra Info **527**. In an embodiment shown in FIG. **8**B, Instruction Set **526** may include or be associated with Extra Info **527**.

Extra Info **527** comprises the functionality for storing any information useful in comparisons or decision making performed in autonomous device operation, and/or other functionalities. One or more Extra Infos **527** can be stored in, appended to, or associated with a Collection of Object Representations **525**, Instruction Set **526**, and/or other element. In some embodiments, the system can obtain Extra Info **527** at a time of creating or generating Collection of Object Representations **525**. In other embodiments, the system can obtain Extra Info **527** at a time of acquiring Instruction Set **526**. In general, Extra Info **527** can be obtained at any time. Examples of Extra Info **527** include time information, location information, computed information, contextual information, and/or other information. Any information can be utilized that can provide information for enhanced comparisons or decision making performed in autonomous device operation. Which information is utilized and/or stored in Extra Info **527** can be set by a user, by DCADO system administrator, or automatically by the system. Extra Info **527** may include or be referred to as contextual information, and vice versa. Therefore, these terms may be used interchangeably herein depending on context.

In some aspects, time information (i.e. time stamp, etc.) can be utilized and/or stored in Extra Info **527**. Time information can be useful in comparisons or decision making performed in autonomous device operation related to a specific time period as Device **98** may be required to perform specific operations at certain parts of day, month, year, and/or other time periods. Time information can be obtained from the system clock, online clock, oscillator, or other time source. In general, Extra Info **527** may include time information related to when Device **98** performed an operation. In other aspects, location information (i.e. coordinates, distance/angle from a known point, address, etc.) can be utilized and/or stored in Extra Info **527**. Location information can be useful in comparisons or decision making performed in autonomous device operation related to a specific place as Device **98** may be required to perform specific operations at certain places. Location information can be obtained from a positioning system (i.e. radio signal triangulation, GPS capabilities, etc.), sensors, and/or other location system. In general, Extra Info **527** may include location information related to where Device **98** performed an operation. In further aspects, computed information can be utilized and/or stored in Extra Info **527**. Computed information can be useful in comparisons or decision making performed in autonomous device operation where information can be calculated, inferred, or derived from other available information. DCADO Unit **100** and/or other disclosed elements may include computational functionalities to create Extra Info **527** by performing calculations or inferences using other information. In one example, Device's **98** speed can be computed or estimated from Device's **98** location and/or time information. In another example, Device's **98** bearing (i.e. angle or direction of movement, etc.) can be computed or estimated from Device's **98** location information by utilizing Pythagorean theorem, trigonometry, and/or other theorems, formulas, or disciplines. In a further example, speeds, bearings, distances, and/or other properties of objects around Device **98** can similarly be computed or inferred using known information. In further aspects, observed information can be utilized and/or stored in Extra Info **527**. In further aspects,

other information can be utilized and/or stored in Extra Info **527**. Examples of such other information include user specific information (i.e. skill level, age, gender, etc.), group user information (i.e. access level, etc.), version of Application Program **18**, type of Application Program **18**, type of Processor **11**, type of Logic Circuit **250**, type of Device **98**, and/or other information.

Referring to FIG. **9**, an embodiment where DCADO Unit **100** is part of or operating on Processor **11** is illustrated. In one example, DCADO Unit **100** may be a hardware element or circuit embedded or built into Processor **11**. In another example, DCADO Unit **100** may be a program operating on Processor **11**.

Referring to FIG. **10**, an embodiment where DCADO Unit **100** resides on Server **96** accessible over Network **95** is illustrated. Any number of Devices **98** may connect to such remote DCADO Unit **100** and the remote DCADO Unit **100** may learn their operations in circumstances including objects with various properties. In turn, any number of Devices **98** can utilize the remote DCADO Unit **100** for autonomous operation in circumstances including objects with various properties. A remote DCADO Unit **100** can be offered as a network service (i.e. online application, etc.). In some aspects, a remote DCADO Unit **100** (i.e. global DCADO Unit **100**, etc.) may reside on the Internet and be available to all the world's Devices **98** configured to transmit their operations in circumstances including objects with various properties and/or configured to utilize the remote DCADO Unit **100** for autonomous operation in circumstances including objects with various properties. For example, multiple operators (i.e. Users **50**, etc.) may operate their Devices **98** where the Devices **98** may be configured to transmit their operations in circumstances including objects with various properties to a remote DCADO Unit **100**. Such remote DCADO Unit **100** enables learning of the operators' collective knowledge of operating Device **98** in circumstances including objects with various properties. Server **96** may be or include any type or form of a remote computing device such as an application server, a network service server, a cloud server, a cloud, and/or other remote computing device. Server **96** may include any features, functionalities, and embodiments of the previously described Computing Device **70**. It should be understood that Server **96** does not have to be a separate computing device and that Server **96**, its elements, or its functionalities can be implemented on Device **98**. Network **95** may include various networks, connection types, protocols, interfaces, APIs, and/or other elements or techniques known in art all of which are within the scope of this disclosure. Any of the previously described networks, network or connection types, networking interfaces, and/or other networking elements or techniques can similarly be utilized. Any of the disclosed elements may reside on Server **96** in alternate implementations. In one example, Artificial Intelligence Unit **110** can reside on Server **96** and Acquisition Interface **120** and/or Modification Interface **130** can reside on Device **98**. In another example, Knowledgebase **530** can reside on Server **96** and the rest of the elements of DCADO Unit **100** can reside on Device **98**. Any other combination of local and remote elements can be implemented.

Referring to FIG. **11**, an embodiment of learning and/or using Remote Device's **97** circumstances for autonomous Remote Device **97** operation is illustrated. In such embodiments, in addition to providing input into Object Processing Unit **93** for learning functionalities herein, Sensor **92** (i.e. Camera **92***a*, Radar **92***d*, Sonar **92***e*, etc.) can provide input into Display **21** or other device for User's **50** perception of

Remote Device's **97** surrounding. As User **50** operates Remote Device **97**, DCADO Unit **100** may learn Remote Device's **97** operation in circumstances including objects with various properties. Such embodiments can be utilized in any situation where one device controls (i.e. remote controls, etc.) another device, any situation where some or all of the processing is on one device and sensor capabilities are on another device, and/or other situations. In one example, a drone controlling device (i.e. Device **98**, etc.) may send control signals to operate a drone (i.e. Remote Device **97**, etc.) and receive information on the drone's surrounding from Sensor **92** on the drone. In another example, a robot controlling device (i.e. Device **98**, etc.) may send control signals to operate a robot (i.e. Remote Device **97**, etc.) and receive information on the robot's surrounding from Sensor **92** on the robot. Any of the disclosed elements in addition to Sensor **92** may reside on Remote Device **97** in alternate implementations.

Referring to FIG. **12**, an embodiment of Artificial Intelligence Unit **110** is illustrated. Artificial Intelligence Unit **110** comprises interconnected Knowledge Structuring Unit **520**, Knowledgebase **530**, Decision-making Unit **540**, and Confirmation Unit **550**. Other additional elements can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate embodiments.

Artificial Intelligence Unit **110** comprises the functionality for learning Device's **98** operation in circumstances including objects with various properties. Artificial Intelligence Unit **110** comprises the functionality for learning one or more collections of object representations correlated with any instruction sets, data, and/or other information. In some aspects, Artificial Intelligence Unit **110** comprises the functionality for learning one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**. In other aspects, Artificial Intelligence Unit **110** comprises the functionality for learning one or more Collections of Object Representations **525** some of which may not be correlated with any Instruction Sets **526** and/or Extra Info **527**. Further, Artificial Intelligence Unit **110** comprises the functionality for anticipating Device's **98** operation in circumstances including objects with various properties. Artificial Intelligence Unit **110** comprises the functionality for anticipating one or more instruction sets, data, and/or other information. Artificial Intelligence Unit **110** comprises the functionality for anticipating one or more Instruction Sets **526** based on one or more incoming Collections of Object Representations **525**. Artificial Intelligence Unit **110** comprises the functionality for anticipating one or more Instruction Sets **526** to be used or executed in Device's **98** autonomous operation. Artificial Intelligence Unit **110** also comprises other disclosed functionalities.

Knowledge Structuring Unit **520**, Knowledgebase **530**, and Decision-making Unit **540** are described later.

Confirmation Unit **550** comprises the functionality for confirming, modifying, evaluating (i.e. rating, etc.), and/or canceling one or more anticipatory Instruction Sets **526**, and/or other functionalities. Confirmation Unit **550** is an optional element that can be omitted depending on implementation. In some embodiments, Confirmation Unit **550** can serve as a means of confirming anticipatory Instruction Sets **526**. For example, Decision-making Unit **540** may determine one or more anticipatory Instruction Sets **526** and provide them to User **50** for confirmation. User **50** may be provided with an interface (i.e. graphical user interface, selectable list of anticipatory Instruction Sets **526**, etc.) to approve or confirm execution of the anticipatory Instruction

Sets **526**. In some aspects, Confirmation Unit **550** can automate User **50** confirmation. In one example, if one or more incoming Collections of Object Representations **525** from Object Processing Unit **93** and one or more Collections of Object Representations **525** from a Knowledge Cell **800** were found to be a perfect or highly similar match, anticipatory Instruction Sets **526** correlated with the one or more Collections of Object Representations **525** from the Knowledge Cell **800** can be automatically executed without User's **50** confirmation. Conversely, if one or more incoming Collections of Object Representations **525** from Object Processing Unit **93** and one or more Collections of Object Representations **525** from a Knowledge Cell **800** were found to be less than a highly similar match, anticipatory Instruction Sets **526** correlated with the one or more Collections of Object Representations **525** from the Knowledge Cell **800** can be presented to User **50** for confirmation and/or modifying. Any features, functionalities, and/or embodiments of Similarity Comparison **125** (later described) can be utilized for such similarity determination. In other embodiments, Confirmation Unit **550** can serve as a means of modifying or editing anticipatory Instruction Sets **526**. For example, Decision-making Unit **540** may determine one or more anticipatory Instruction Sets **526** and provide them to User **50** for modification. User **50** may be provided with an interface (i.e. graphical user interface, etc.) to modify the anticipatory Instruction Sets **526** before their execution. In further embodiments, Confirmation Unit **550** can serve as a means of evaluating or rating anticipatory Instruction Sets **526** if they matched User's **50** intended operation. For example, Decision-making Unit **540** may determine one or more anticipatory Instruction Sets **526**, which the system may automatically execute. User **50** may be provided with an interface (i.e. graphical user interface, etc.) to rate (i.e. on a scale from 0 to 1, etc.) how well Decision-making Unit **540** predicted the executed anticipatory Instruction Sets **526**. In some aspects, rating can be automatic and based on a particular function or method that rates how well the anticipatory Instruction Sets **526** matched the desired operation. In one example, a rating function or method can assign a higher rating to anticipatory Instruction Sets **526** that were least modified in the confirmation process. In another example, a rating function or method can assign a higher rating to anticipatory Instruction Sets **526** that were canceled least number of times by User **50**. Any other automatic rating function or method can be utilized. In yet other embodiments, Confirmation Unit **550** can serve as a means of canceling anticipatory Instruction Sets **526** if they did not match User's **50** intended operation. For example, Decision-making Unit **540** may determine one or more anticipatory Instruction Sets **526**, which the system may automatically execute. The system may save the state of Device **98**, Processor **11** (save its register values, etc.), Logic Circuit **250**, Application Program **18** (i.e. save its variables, data structures, objects, location of its current instruction, etc.), and/or other processing elements before executing anticipatory Instruction Sets **526**. User **50** may be provided with an interface (i.e. graphical user interface, selectable list of prior executed anticipatory Instruction Sets **526**, etc.) to cancel one or more of the prior executed anticipatory Instruction Sets **526**, and restore Device **98**, Processor **11**, Logic Circuit **250**, Application Program **18**, and/or other processing elements to a prior state. In some aspects, Confirmation Unit **550** can optionally be disabled or omitted in order to provide an uninterrupted operation of Device **98**, Processor **11**, Logic Circuit **250**, and/or Application Program **18**. For example, a thermostat may be suitable for implementing the

user confirmation step, whereas, a vehicle may be less suitable for implementing such interrupting step due to the real time nature of vehicle operation.

Referring to FIG. **13**, an embodiment of Knowledge Structuring Unit **520** correlating individual Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527** is illustrated. Knowledge Structuring Unit **520** comprises the functionality for structuring the knowledge of a device's operation in circumstances including objects with various properties, and/or other functionalities. Knowledge Structuring Unit **520** comprises the functionality for correlating one or more Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527**. Knowledge Structuring Unit **520** comprises the functionality for creating or generating Knowledge Cell **800** and storing one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** into the Knowledge Cell **800**. As such, Knowledge Cell **800** comprises the functionality for storing one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**. Knowledge Cell **800** includes knowledge (i.e. unit of knowledge, etc.) of how Device **98** operated in a circumstance including objects with various properties. Once created or generated, Knowledge Cells **800** can be used in/as neurons, nodes, vertices, or other elements in any of the data structures or arrangements (i.e. neural networks, graphs, sequences, etc.) used for storing the knowledge of Device's **98** operation in circumstances including objects with various properties, thereby facilitating learning functionalities herein. It should be noted that Extra Info **527** may be optionally used in some implementations to enable enhanced comparisons or decision making in autonomous device operation where applicable, and that Extra Info **527** can be omitted in alternate implementations.

In some embodiments, Knowledge Structuring Unit **520** receives one or more Collections of Object Representations **525** from Object Processing Unit **93**. Knowledge Structuring Unit **520** may also receive one or more Instruction Sets **526** from Acquisition Interface **120**. Knowledge Structuring Unit **520** may further receive any Extra Info **527**. Although, Extra Info **527** is not shown in this and/or other figures for clarity of illustration, it should be noted that any Collection of Object Representations **525**, Instruction Set **526**, and/or other element may include or be associated with Extra Info **527**. Knowledge Structuring Unit **520** may correlate one or more Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527**. Knowledge Structuring Unit **520** may then create Knowledge Cell **800** and store the one or more Collections of Object Representations **525** correlated with Instruction Sets **526** and/or Extra Info **527** into the Knowledge Cell **800**. Knowledge Cell **800** may include any data structure or arrangement that can facilitate such storing. For example, Knowledge Structuring Unit **520** may create Knowledge Cell **800***ax* and structure within it Collection of Object Representations **525***a***1** correlated with Instruction Sets **526***a***1**-**526***a***3** and/or any Extra Info **527** (not shown). Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800***ax* a Collection of Object Representations **525***a***2** correlated with Instruction Set **526***a***4** and/or any Extra Info **527** (not shown). Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800***ax* a Collection of Object Representations **525***a***3** without a correlated Instruction Set **526** and/or Extra Info **527**. Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800***ax* a Collection of Object Representations **525***a***4** correlated with Instruction

Sets **526**_a_**5**-**526**_a_**6** and/or any Extra Info **527** (not shown). Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800**_ax_ a Collection of Object Representations **525**_a_**5** without a correlated Instruction Set **526** and/or Extra Info **527**. Knowledge Structuring Unit **520** may structure within Knowledge Cell **800**_ax_ additional Collections of Object Representations **525** correlated with any number (including zero [i.e. uncorrelated]) of Instruction Sets **526** and/or Extra Info **527** by following similar logic as described above.

In some embodiments, Knowledge Structuring Unit **520** may correlate a Collection of Object Representations **525** with one or more temporally corresponding Instruction Sets **526** and/or Extra Info **527**. This way, Knowledge Structuring Unit **520** can structure the knowledge of Device's **98** operation at or around the time of generating Collections of Object Representations **525**. Such functionality enables spontaneous or seamless learning of Device's **98** operation in circumstances including objects with various properties as Device **98** is operated in real life situations. In some designs, Knowledge Structuring Unit **520** may receive a stream of Instruction Sets **526** used or executed to effect Device's **98** operations as well as a stream of Collections of Object Representations **525** as the operations are performed. Knowledge Structuring Unit **520** can then correlate Collections of Object Representations **525** from the stream of Collections of Object Representations **525** with temporally corresponding Instruction Sets **526** from the stream of Instruction Sets **526** and/or any Extra Info **527**. Collections of Object Representations **525** without a temporally corresponding Instruction Set **526** may be uncorrelated, for instance. In some aspects, Instruction Sets **526** and/or Extra Info **527** that temporally correspond to a Collection of Object Representations **525** may include Instruction Sets **526** used and/or Extra Info **527** obtained at the time of generating the Collection of Object Representations **525**. In other aspects, Instruction Sets **526** and/or Extra Info **527** that temporally correspond to a Collection of Object Representations **525** may include Instruction Sets **526** used and/or Extra Info **527** obtained within a certain time period before and/or after generating the Collection of Object Representations **525**. For example, Instruction Sets **526** and/or Extra Info **527** that temporally correspond to a Collection of Object Representations **525** may include Instruction Sets **526** used and/or Extra Info **527** obtained within 50 milliseconds, 1 second, 3 seconds, 20 seconds, 1 minute, 41 minutes, 2 hours, or any other time period before and/or after generating the Collection of Object Representations **525**. Such time periods can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. In other aspects, Instruction Sets **526** and/or Extra Info **527** that temporally correspond to a Collection of Object Representations **525** may include Instruction Sets **526** used and/or Extra Info **527** obtained from the time of generating the Collection of Object Representations **525** to the time of generating a next Collection of Object Representations **525**. In further aspects, Instruction Sets **526** and/or Extra Info **527** that temporally correspond to a Collection of Object Representations **525** may include Instruction Sets **526** used and/or Extra Info **527** obtained from the time of generating a previous Collection of Object Representations **525** to the time of generating the Collection of Object Representations **525**. Any other temporal relationship or correspondence between Collections of Object Representations **525** and correlated Instruction Sets **526** and/or Extra Info **527** can be implemented.

In some embodiments, Knowledge Structuring Unit **520** can structure the knowledge of Device's **98** operation in a circumstance including objects with various properties into any number of Knowledge Cells **800**. In some aspects, Knowledge Structuring Unit **520** can structure into a Knowledge Cell **800** a single Collection of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**. In other aspects, Knowledge Structuring Unit **520** can structure into a Knowledge Cell **800** any number (i.e. 2, 4, 7, 17, 29, 87, 1415, 23891, 323674, 8132401, etc.) of Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**. In a special case, Knowledge Structuring Unit **520** can structure all Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** into a single long Knowledge Cell **800**. In further aspects, Knowledge Structuring Unit **520** can structure Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** into a plurality of Knowledge Cells **800**. In a special case, Knowledge Structuring Unit **520** can store periodic streams of Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** into a plurality of Knowledge Cells **800** such as hourly, daily, weekly, monthly, yearly, or other periodic Knowledge Cells **800**.

In some embodiments, Device **98** may include a plurality of Sensors **92** and/or their corresponding Object Processing Units **93**. In one example, multiple Sensors **92** may detect objects and/or their properties from different angles or on different sides of Device **98**. In another example, one or more Sensors **92** may be placed on different sub-devices, sub-systems, or elements of Device **98**. Using multiple Sensors **92** and/or their corresponding Object Processing Units **93** may provide additional detail in learning and/or using Device's **98** circumstances for autonomous Device **98** operation. In some designs where multiple Sensors **92** and/or their corresponding Object Processing Units **93** are utilized, multiple DCADO Units **100** can also be utilized (i.e. one DCADO Unit **100** for each Sensor **92** and its corresponding Object Processing Unit **93**, etc.). In such designs, Collections of Object Representations **525** can be correlated with any Instruction Sets **526** and/or Extra Info **527** as previously described. In other designs where multiple Sensors **92** and/or their corresponding Object Processing Units **93** are utilized, collective Collections of Object Representations **525** from multiple Sensors **92** and their corresponding Object Processing Units **93** can be correlated with any Instruction Sets **526** and/or Extra Info **527**.

In some embodiments, Device **98** may include a plurality of Logic Circuits **250**, Processors **11**, Application Programs **18**, and/or other processing elements. For example, each processing element may control a sub-device, sub-system, or an element of Device **98**. Using multiple processing elements may provide enhanced control over Device's **98** operation. In some designs where multiple processing elements are utilized, multiple DCADO Units **100** can also be utilized (i.e. one DCADO Unit **100** for each processing element, etc.). In such designs, Collections of Object Representations **525** can be correlated with any Instruction Sets **526** and/or Extra Info **527** as previously described. In other designs where multiple processing elements are utilized, Collections of Object Representations **525** can be correlated with any collective Instruction Sets **526** and/or Extra Info **527** used or executed by a plurality of processing elements.

Any combination of the aforementioned multiple Sensors **92** and/or their corresponding Object Processing Units **93**,

multiple processing elements, and/or other elements can be implemented in alternate embodiments.

Referring to FIG. **14**, another embodiment of Knowledge Structuring Unit **520** correlating individual Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527** is illustrated. In such embodiments, Knowledge Structuring Unit **520** may generate Knowledge Cells **800** each comprising a single Collection of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**.

Referring to FIG. **15**, an embodiment of Knowledge Structuring Unit **520** correlating streams of Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527** is illustrated. For example, Knowledge Structuring Unit **520** may create Knowledge Cell **800*ax*** and structure within it a stream of Collections of Object Representations **525*a*1-525*an*** correlated with Instruction Set **526*a*1** and/or any Extra Info **527** (not shown). Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800*ax*** a stream of Collections of Object Representations **525*b*1-525*bn*** correlated with Instruction Sets **526*a*2-526*a*4** and/or and Extra Info **527** (not shown). Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800*ax*** a stream of Collections of Object Representations **525*c*1-525*cn*** without correlated Instruction Sets **526** and/or Extra Info **527**. Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800*ax*** a stream of Collections of Object Representations **525*d*1-525*dn*** correlated with Instruction Sets **526*a*5-526*a*6** and/or any Extra Info **527** (not shown). Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800*ax*** additional streams of Collections of Object Representations **525** correlated with any number (including zero [i.e. uncorrelated]) of Instruction Sets **526** and/or Extra Info **527** by following similar logic as described above. The number of Collections of Object Representations **525** in some or all streams of Collections of Object Representations **525*a*1-525*an***, **525*b*1-525*bn***, etc. may be equal or different. It should be noted that n or other such letters or indicia may follow the sequence and/or context where they are indicated. Also, a same letter or indicia such as n may represent a different number in different elements of a drawing.

Referring to FIG. **16**, another embodiment of Knowledge Structuring Unit **520** correlating streams of Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527** is illustrated. In such embodiments, Knowledge Structuring Unit **520** may generate Knowledge Cells **800** each comprising a single stream of Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**.

Knowledgebase **530** comprises the functionality for storing the knowledge of a device's operation in circumstances including objects with various properties, and/or other functionalities. Knowledgebase **530** comprises the functionality for storing one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**. Knowledgebase **530** comprises the functionality for storing one or more Knowledge Cells **800** each including one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**. In some aspects, Collections of Object Representations **525** correlated with Instruction Sets **526** and/or Extra Info **527** can be stored directly within Knowledgebase **530** without using Knowledge Cells **800** as the intermediary data structures. In some embodiments, Knowledgebase **530** may be or include Neural Network **530*a*** (later described). In other embodiments, Knowledgebase **530** may be or include Graph **530*b*** (later described).

In further embodiments, Knowledgebase **530** may be or include Collection of Sequences **530*c*** (later described). In further embodiments, Knowledgebase **530** may be or include Sequence **533** (later described). In further embodiments, Knowledgebase **530** may be or include Collection of Knowledge Cells **530*d*** (later described). In general, Knowledgebase **530** may be or include any data structure or arrangement capable of storing the knowledge of a device's operation in circumstances including objects with various properties. Knowledgebase **530** may reside locally on Device **98**, or remotely (i.e. remote Knowledgebase **530**, etc.) on a remote computing device (i.e. server, cloud, etc.) accessible over a network or an interface.

In some embodiments, Knowledgebase **530** from one Device **98** or DCADO Unit **100** can be transferred to one or more other Devices **98** or DCADO Units **100**. Therefore, the knowledge of Device's **98** operation in circumstances including objects with various properties learned on one Device **98** or DCADO Unit **100** can be transferred to one or more other Devices **98** or DCADO Units **100**. In one example, Knowledgebase **530** can be copied or downloaded to a file or other repository from one Device **98** or DCADO Unit **100** and loaded or inserted into another Device **98** or DCADO Unit **100**. In another example, Knowledgebase **530** from one Device **98** or DCADO Unit **100** can be available on a server accessible by other Devices **98** or DCADO Units **100** over a network or an interface. Once loaded into or accessed by a receiving Device **98** or DCADO Unit **100**, the receiving Device **98** or DCADO Unit **100** can then implement the knowledge of Device's **98** operation in circumstances including objects with various properties learned on the originating Device **98** or DCADO Unit **100**.

In some embodiments, multiple Knowledgebases **530** (i.e. Knowledgebases **530** from different Devices **98** or DCADO Units **100**, etc.) can be combined to accumulate collective knowledge of operating Device **98** in circumstances including objects with various properties. In one example, one Knowledgebase **530** can be appended to another Knowledgebase **530** such as appending one Collection of Sequences **530*c*** (later described) to another Collection of Sequences **530*c***, appending one Sequence **533** (later described) to another Sequence **533**, appending one Collection of Knowledge Cells **530*d*** (later described) to another Collection of Knowledge Cells **530*d***, and/or appending other data structures or elements thereof. In another example, one Knowledgebase **530** can be copied into another Knowledgebase **530** such as copying one Collection of Sequences **530*c*** into another Collection of Sequences **530*c***, copying one Collection of Knowledge Cells **530*d*** into another Collection of Knowledge Cells **530*d***, and/or copying other data structures or elements thereof. In a further example, in the case of Knowledgebase **530** being or including Graph **530*b*** or graph-like data structure (i.e. Neural Network **530*a***, tree, etc.), a union can be utilized to combine two or more Graphs **530*b*** or graph-like data structures. For instance, a union of two Graphs **530*b*** or graph-like data structures may include a union of their vertex (i.e. node, etc.) sets and their edge (i.e. connection, etc.) sets. Any other operations or combination thereof on graphs or graph-like data structures can be utilized to combine Graphs **530*b*** or graph-like data structures. In a further example, one Knowledgebase **530** can be combined with another Knowledgebase **530** through later described learning processes where Knowledge Cells **800** may be applied one at a time and connected with prior and/or subsequent Knowledge Cells

800 such as in Graph **530***b* or Neural Network **530***a*. In such embodiments, instead of Knowledge Cells **800** generated by Knowledge Structuring Unit **520**, the learning process may utilize Knowledge Cells **800** from one Knowledgebase **530** to apply them onto another Knowledgebase **530**. Any other techniques known in art including custom techniques for combining data structures can be utilized for combining Knowledgebases **530** in alternate implementations. In any of the aforementioned and/or other combining techniques, similarity of elements (i.e. nodes/vertices, edges/connections, etc.) can be utilized in determining whether an element from one Knowledgebase **530** matches an element from another Knowledgebase **530**, and substantially or otherwise similar elements may be considered a match for combining purposes in some designs. Any features, functionalities, and embodiments of Similarity Comparison **125** (later described) can be used in such similarity determinations. A combined Knowledgebase **530** can be offered as a network service (i.e. online application, etc.), downloadable file, or other repository to all DCADO Units **100** configured to utilize the combined Knowledgebase **530**. For example, a Device **98** including or interfaced with DCADO Unit **100** having access to a combined Knowledgebase **530** can use the collective knowledge learned from multiple Devices **98** for the Device's **98** autonomous operation.

Referring to FIG. **17**, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation may include various artificial intelligence models and/or techniques. The disclosed systems, devices, and methods are independent of the artificial intelligence model and/or technique used and any model and/or technique can be utilized to facilitate the functionalities described herein. Examples of these models and/or techniques include deep learning, supervised learning, unsupervised learning, neural networks (i.e. convolutional neural network, recurrent neural network, deep neural network, etc.), search-based, logic and/or fuzzy logic-based, optimization-based, tree/graph/other data structure-based, hierarchical, symbolic and/or sub-symbolic, evolutionary, genetic, multi-agent, deterministic, probabilistic, statistical, and/or other models and/or techniques.

In one example shown in Model A, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation may include a neural network (also referred to as artificial neural network, etc.). As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include a network of Nodes **852** (also referred to as neurons, etc.) and Connections **853** similar to that of a brain. Node **852** can store any data, object, data structure, and/or other item, or reference thereto. Node **852** may also include a function for transforming or manipulating any data, object, data structure, and/or other item. Examples of such transformation functions include mathematical functions (i.e. addition, subtraction, multiplication, division, sin, cos, log, derivative, integral, etc.), object manipulation functions (i.e. creating an object, modifying an object, deleting an object, appending objects, etc.), data structure manipulation functions (i.e. creating a data structure, modifying a data structure, deleting a data structure, creating a data field, modifying a data field, deleting a data field, etc.), and/or other transformation functions. Connection **853** may include or be associated with a value such as a symbolic label or numeric attribute (i.e. weight, cost, capacity, length, etc.). A computational model can be utilized to compute values from inputs based on a pre-programmed or learned function or method. For example, a neural network may include one or more input neurons that can be activated by inputs. Activations of these neurons can then be passed on, weighted, and transformed by a function to other neurons. Neural networks may range from those with only one layer of single direction logic to multi-layer of multi-directional feedback loops. A neural network can use weights to change the parameters of the network's throughput. A neural network can learn by input from its environment or from self-teaching using written-in rules. A neural network can be utilized as a predictive modeling approach in machine learning. An exemplary embodiment of a neural network (i.e. Neural Network **530***a*, etc.) is described later.

In another example shown in Model B, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation may include a graph or graph-like data structure. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include Nodes **852** (also referred to as vertices or points, etc.) and Connections **853** (also referred to as edges, arrows, lines, arcs, etc.) organized as a graph. In general, any Node **852** in a graph can be connected to any other Node **852**. A Connection **853** may include unordered pair of Nodes **852** in an undirected graph or ordered pair of Nodes **852** in a directed graph. Nodes **852** can be part of the graph structure or external entities represented by indices or references. A graph can be utilized as a predictive modeling approach in machine learning. Nodes **852**, Connections **853**, and/or other elements or operations of a graph may include any features, functionalities, and embodiments of the aforementioned Nodes **852**, Connections **853**, and/or other elements or operations of a neural network, and vice versa. An exemplary embodiment of a graph (i.e. Graph **530***b*, etc.) is described later.

In a further example shown in Model C, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation may include a tree or tree-like data structure. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include Nodes **852** and Connections **853** (also referred to as references, edges, etc.) organized as a tree. In general, a Node **852** in a tree can be connected to any number (i.e. including zero, etc.) of children Nodes **852**. A tree can be utilized as a predictive modeling approach in machine learning. Nodes **852**, Connections **853**, and/or other elements or operations of a tree may include any features, functionalities, and embodiments of the aforementioned Nodes **852**, Connections **853**, and/or other elements or operations of a neural network and/or graph, and vice versa.

In a further example shown in Model D, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation may include a sequence or sequence-like data structure. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include a structure of Nodes **852** and/or Connections **853** organized as a sequence. In some aspects, Connections **853** may be optionally omitted from a sequence as the sequential order of Nodes **852** in a sequence may be implied in the structure. A sequence can be utilized as a predictive modeling approach in machine learning. Nodes **852**, Connections **853**, and/or other elements or operations of a

sequence may include any features, functionalities, and embodiments of the aforementioned Nodes **852**, Connections **853**, and/or other elements or operations of a neural network, graph, and/or tree, and vice versa. An exemplary embodiment of a sequence (i.e. Collection of Sequences **530***c*, Sequence **533**, etc.) is described later.

In yet another example, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation may include a search-based model and/or technique. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include searching through a collection of possible solutions. For example, a search method can search through a neural network, graph, tree, sequence, or other data structure that includes data elements of interest. A search may use heuristics to limit the search for solutions by eliminating choices that are unlikely to lead to the goal. Heuristic techniques may provide a best guess solution. A search can also include optimization. For example, a search may begin with a guess and then refine the guess incrementally until no more refinements can be made. In a further example, the disclosed systems, devices, and methods may include logic-based model and/or technique. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities can use formal or other type of logic. Logic based models may involve making inferences or deriving conclusions from a set of premises. As such, a logic based system can extend existing knowledge or create new knowledge automatically using inferences. Examples of the types of logic that can be utilized include propositional or sentential logic that comprises logic of statements which can be true or false; first-order logic that allows the use of quantifiers and predicates and that can express facts about objects, their properties, and their relations with each other; fuzzy logic that allows degrees of truth to be represented as a value between 0 and 1 rather than simply 0 (false) or 1 (true), which can be used for uncertain reasoning; subjective logic that comprises a type of probabilistic logic that may take uncertainty and belief into account, which can be suitable for modeling and analyzing situations involving uncertainty, incomplete knowledge and different world views; and/or other types of logic. In a further example, the disclosed systems, devices, and methods may include a probabilistic model and/or technique. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities can be implemented to operate with incomplete or uncertain information where probabilities may affect outcomes. Bayesian network, among other models, is an example of a probabilistic tool used for purposes such as reasoning, learning, planning, perception, and/or others. One of ordinary skill in art will understand that the aforementioned artificial intelligence models and/or techniques are described merely as examples of a variety of possible implementations, and that while all possible artificial intelligence models and/or techniques are too voluminous to describe, other artificial intelligence models and/or techniques known in art are within the scope of this disclosure. One of ordinary skill in art will also recognize that an intelligent system may solve a specific problem by using any model and/or technique that works such as, for example, some systems can be symbolic and logical, some can be sub-symbolic neural networks, some can be deterministic or probabilistic, some can be hierarchical, some may include searching techniques, some may include optimization techniques, while others may use other or a combination of models and/or techniques. In general, any artificial intelligence model and/or technique can be utilized that can facilitate the functionalities described herein.

Referring to FIGS. **18**A-**18**C, embodiments of interconnected Knowledge Cells **800** and updating weights of Connections **853** are illustrated. As shown for example in FIG. **18**A, Knowledge Cell **800***za* is connected to Knowledge Cell **800***zb* and Knowledge Cell **800***zc* by Connection **853***z***1** and Connection **853***z***2**, respectively. Each of Connection **853***z***1** and Connection **853***z***2** may include or be associated with occurrence count, weight, and/or other parameter or data. The number of occurrences may track or store the number of observations that a Knowledge Cell **800** was followed by another Knowledge Cell **800** indicating a connection or relationship between them. For example, Knowledge Cell **800***za* was followed by Knowledge Cell **800***zb* 10 times as indicated by the number of occurrences of Connection **853***z***1**. Also, Knowledge Cell **800***za* was followed by Knowledge Cell **800***zc* 15 times as indicated by the number of occurrences of Connection **853***z***2**. The weight of Connection **853***z***1** can be calculated or determined as the number of occurrences of Connection **853***z***1** divided by the sum of occurrences of all connections (i.e. Connection **853***z***1** and Connection **853***z***2**, etc.) originating from Knowledge Cell **800***za*. Therefore, the weight of Connection **853***z***1** can be calculated or determined as 10/(10+15)=0.4, for example. Also, the weight of Connection **853***z***2** can be calculated or determined as 15/(10+15)=0.6, for example. Therefore, the sum of weights of Connection **853***z***1**, Connection **853***z***2**, and/or any other Connections **853** originating from Knowledge Cell **800***za* may equal to 1 or 100%. As shown for example in FIG. **18**B, in the case that Knowledge Cell **800***zd* is inserted and an observation is made that Knowledge Cell **800***zd* follows Knowledge Cell **800***za*, Connection **853***z***3** can be created between Knowledge Cell **800***za* and Knowledge Cell **800***zd*. The occurrence count of Connection **853***z***3** can be set to 1 and weight determined as 1/(10+15+1)=0.038. The weights of all other connections (i.e. Connection **853***z***1**, Connection **853***z***2**, etc.) originating from Knowledge Cell **800***za* may be updated to account for the creation of Connection **853***z***3**. Therefore, the weight of Connection **853***z***1** can be updated as 10/(10+15+1)=0.385. The weight of Connection **853***z***2** can also be updated as 15/(10+15+1)=0.577. As shown for example in FIG. **18**C, in the case that an additional occurrence of Connection **853***z***1** is observed (i.e. Knowledge Cell **800***zb* followed Knowledge Cell **800***za*, etc.), occurrence count of Connection **853***z***1** and weights of all connections (i.e. Connection **853***z***1**, Connection **853***z***2**, and Connection **853***z***3**, etc.) originating from Knowledge Cell **800***za* may be updated to account for this observation. The occurrence count of Connection **853***z***1** can be increased by 1 and its weight updated as 11/(11+15+1)=0.407. The weight of Connection **853***z***2** can also be updated as 15/(11+15+1)=0.556. The weight of Connection **853***z***3** can also be updated as 1/(11+15+1)=0.037.

Referring to FIG. **19**, an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Collection of Knowledge Cells **530***d* is illustrated. Collection of Knowledge Cells **530***d* comprises the functionality for storing any number of Knowledge Cells **800**. In some aspects, Knowledge Cells **800** may be stored into or applied onto Collection of Knowledge Cells **530***d* in a learning or training process. In effect, Collection of Knowledge Cells **530***d* may store

Knowledge Cells **800** that can later be used to enable autonomous Device **98** operation. In some embodiments, Knowledge Structuring Unit **520** structures or generates Knowledge Cells **800** as previously described and the system applies them onto Collection of Knowledge Cells **530***d*, thereby implementing learning Device's **98** operation in circumstances including objects with various properties. The term apply or applying may refer to storing, copying, inserting, updating, or other similar action, therefore, these terms may be used interchangeably herein depending on context. The system can perform Similarity Comparisons **125** (later described) of a newly structured Knowledge Cell **800** from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. If a substantially similar Knowledge Cell **800** is not found in Collection of Knowledge Cells **530***d*, the system may insert (i.e. copy, store, etc.) the Knowledge Cell **800** from Knowledge Structuring Unit **520** into Collection of Knowledge Cells **530***d*, for example. On the other hand, if a substantially similar Knowledge Cell **800** is found in Collection of Knowledge Cells **530***d*, the system may optionally omit inserting the Knowledge Cell **800** from Knowledge Structuring Unit **520** as inserting a substantially similar Knowledge Cell **800** may not add much or any additional knowledge to the Collection of Knowledge Cells **530***d*, for example. Also, inserting a substantially similar Knowledge Cell **800** can optionally be omitted to save storage resources and limit the number of Knowledge Cells **800** that may later need to be processed or compared. Any features, functionalities, and embodiments of Similarity Comparison **125**, importance index (later described), similarity index (later described), and/or other disclosed elements can be utilized to facilitate determination of substantial or other similarity and whether to insert a newly structured Knowledge Cell **800** into Collection of Knowledge Cells **530***d*.

For example, the system can perform Similarity Comparisons **125** (later described) of Knowledge Cell **800***ba* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. In the case that a substantially similar match is found between Knowledge Cell **800***ba* and any of the Knowledge Cells **800** in Collection of Knowledge Cells **530***d*, the system may perform no action. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bb* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. In the case that a substantially similar match is not found, the system may insert a new Knowledge Cell **800** into Collection of Knowledge Cells **530***d* and copy Knowledge Cell **800***bb* into the inserted new Knowledge Cell **800**. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bc* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. In the case that a substantially similar match is found between Knowledge Cell **800***bc* and any of the Knowledge Cells **800** in Collection of Knowledge Cells **530***d*, the system may perform no action. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bd* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. In the case that a substantially similar match is not found, the system may insert a new Knowledge Cell **800** into Collection of Knowledge Cells **530***d* and copy Knowledge Cell **800***bd* into the inserted new Knowledge Cell **800**. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***be* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. In the case that a substantially similar match is not found, the system may insert a new Knowledge Cell **800** into Collection of Knowledge Cells **530***d* and copy Knowledge Cell **800***be* into the inserted new Knowledge Cell **800**. Applying any additional Knowledge Cells **800** from Knowledge Structuring Unit **520** onto Collection of Knowledge Cells **530***d* follows similar logic or process as the above-described.

Referring to FIG. **20**, an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Neural Network **530***a* is illustrated. Neural Network **530***a* includes a number of neurons or Nodes **852** interconnected by Connections **853** as previously described. Knowledge Cells **800** are shown instead of Nodes **852** to simplify the illustration as Node **852** includes a Knowledge Cell **800**, for example. Therefore, Knowledge Cells **800** and Nodes **852** can be used interchangeably herein depending on context. It should be noted that Node **852** may include other elements and/or functionalities instead of or in addition to Knowledge Cell **800**. In some aspects, Knowledge Cells **800** may be stored into or applied onto Neural Network **530***a* individually or collectively in a learning or training process. In some designs, Neural Network **530***a* comprises a number of Layers **854** each of which may include one or more Knowledge Cells **800**. Knowledge Cells **800** in successive Layers **854** can be connected by Connections **853**. Connection **853** may include or be associated with occurrence count, weight, and/or other parameter or data as previously described. Neural Network **530***a* may include any number of Layers **854** comprising any number of Knowledge Cells **800**. In some aspects, Neural Network **530***a* may store Knowledge Cells **800** interconnected by Connections **853** where following a path through the Neural Network **530***a* can later be used to enable autonomous Device **98** operation. It should be understood that, in some embodiments, Knowledge Cells **800** in one Layer **854** of Neural Network **530***a* need not be connected only with Knowledge Cells **800** in a successive Layer **854**, but also in any other Layer **854**, thereby creating shortcuts (i.e. shortcut Connections **853**, etc.) through Neural Network **530***a*. A Knowledge Cell **800** can also be connected to itself such as, for example, in recurrent neural networks. In general, any Knowledge Cell **800** can be connected with any other Knowledge Cell **800** anywhere else in Neural Network **530***a*. In further embodiments, back-propagation of any data or information can be implemented. In one example, back-propagation of similarity (i.e. similarity index, etc.) of compared Knowledge Cells **800** in a path through Neural Network **530***a* can be implemented. In another example, back-propagation of errors can be implemented. Such back-propagations can then be used to adjust occurrence counts and/or weights of Connections **853** for better future predictions, for example. Any other back-propagation can be implemented for other purposes. Any combination of Nodes **852** (i.e. Nodes **852** comprising Knowledge Cells **800**, etc.), Connections **853**, Layers **854**, and/or other elements or techniques can be implemented in alternate embodiments. Neural Network **530***a* may include any type or form of a neural network known in art such as a feed-forward neural network, a back-propagating neural network, a recurrent neural network, a convolutional neural network, deep neural network, and/or others including a custom neural network.

In some embodiments, Knowledge Structuring Unit **520** structures or generates Knowledge Cells **800** and the system applies them onto Neural Network **530***a*, thereby implementing learning Device's **98** operation in circumstances including objects with various properties. The system can

perform Similarity Comparisons **125** (later described) of a Knowledge Cell **800** from Knowledge Structuring Unit **520** with Knowledge Cells **800** in a Layer **854** of Neural Network **530***a*. If a substantially similar Knowledge Cell **800** is not found in the Layer **854** of Neural Network **530***a*, the system may insert (i.e. copy, store, etc.) the Knowledge Cell **800** from Knowledge Structuring Unit **520** into the Layer **854** of Neural Network **530***a*, and create a Connection **853** to the inserted Knowledge Cell **800** from a Knowledge Cell **800** in a prior Layer **854** including assigning an occurrence count to the new Connection **853**, calculating a weight of the new Connection **853**, and updating any other Connections **853** originating from the Knowledge Cell **800** in the prior Layer **854**. On the other hand, if a substantially similar Knowledge Cell **800** is found in the Layer **854** of Neural Network **530***a*, the system may update occurrence count and weight of Connection **853** to that Knowledge Cell **800** from a Knowledge Cell **800** in a prior Layer **854**, and update any other Connections **853** originating from the Knowledge Cell **800** in the prior Layer **854**.

For example, the system can perform Similarity Comparisons **125** (later described) of Knowledge Cell **800***ba* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Layer **854***a* of Neural Network **530***a*. In the case that a substantially similar match is found between Knowledge Cell **800***ba* and Knowledge Cell **800***ea*, the system may perform no action since Knowledge Cell **800***ea* is the initial Knowledge Cell **800**. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bb* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Layer **854***b* of Neural Network **530***a*. In the case that a substantially similar match is found between Knowledge Cell **800***bb* and Knowledge Cell **800***eb*, the system may update occurrence count and weight of Connection **853***e***1** between Knowledge Cell **800***ea* and Knowledge Cell **800***eb*, and update weights of other Connections **853** originating from Knowledge Cell **800***ea* as previously described. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bc* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Layer **854***c* of Neural Network **530***a*. In the case that a substantially similar match is not found, the system may insert Knowledge Cell **800***ec* into Layer **854***c* and copy Knowledge Cell **800***bc* into the inserted Knowledge Cell **800***ec*. The system may also create Connection **853***e***2** between Knowledge Cell **800***eb* and Knowledge Cell **800***ec* with occurrence count of 1 and weight calculated based on the occurrence count as previously described. The system may also update weights of other Connections **853** (one in this example) originating from Knowledge Cell **800***eb* as previously described. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bd* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Layer **854***d* of Neural Network **530***a*. In the case that a substantially similar match is not found, the system may insert Knowledge Cell **800***ed* into Layer **854***d* and copy Knowledge Cell **800***bd* into the inserted Knowledge Cell **800***ed*. The system may also create Connection **853***e***3** between Knowledge Cell **800***ec* and Knowledge Cell **800***ed* with occurrence count of 1 and weight of 1. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***be* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Layer **854***e* of Neural Network **530***a*. In the case that a substantially similar match is not found, the system may insert Knowledge Cell **800***ee* into Layer **854***e* and copy Knowledge Cell **800***be* into the inserted Knowledge Cell **800***ee*. The system may also create Connection **853***e***4** between

Knowledge Cell **800***ed* and Knowledge Cell **800***ee* with occurrence count of 1 and weight of 1. Applying any additional Knowledge Cells **800** from Knowledge Structuring Unit **520** onto Neural Network **530***a* follows similar logic or process as the above-described.

Referring now to Similarity Comparison **125**, Similarity Comparison **125** comprises the functionality for comparing or matching Knowledge Cells **800** or portions thereof, and/or other functionalities. Similarity Comparison **125** comprises the functionality for comparing or matching Collections of Object Representations **525** or portions thereof. Similarity Comparison **125** comprises the functionality for comparing or matching streams of Collections of Object Representations **525** or portions thereof. Similarity Comparison **125** comprises the functionality for comparing or matching Object Representations **625** or portions thereof. Similarity Comparison **125** comprises the functionality for comparing or matching Object Properties **630** or portions thereof. Similarity Comparison **125** comprises the functionality for comparing or matching Instruction Sets **526**, Extra Info **527**, text (i.e. characters, words, phrases, etc.), numbers, and/or other elements or portions thereof. Similarity Comparison **125** may include functions, rules, and/or logic for performing matching or comparisons and for determining that while a perfect match is not found, a partial or similar match has been found. In some aspects, a partial match may include a substantially or otherwise similar match, and vice versa. Therefore, these terms may be used interchangeably herein depending on context. As such, Similarity Comparison **125** may include determining substantial similarity or substantial match of compared elements. Although, substantial similarity or substantial match is frequently used herein, it should be understood that any level of similarity, however high or low, may be utilized as defined by the rules (i.e. thresholds, etc.) for similarity. The rules for similarity or similar match can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. In some designs, Similarity Comparison **125** comprises the functionality to automatically define appropriately strict rules for determining similarity of the compared elements. Similarity Comparison **125** can therefore set, reset, and/or adjust the strictness of the rules for finding or determining similarity of the compared elements, thereby fine tuning Similarity Comparison **125** so that the rules for determining similarity are appropriately strict. In some aspects, the rules for determining similarity may include a similarity threshold. As such, Similarity Comparison **125** can determine similarity of compared elements if their similarity exceeds a similarity threshold. In other aspects, the rules for determining similarity may include a difference threshold. As such, Similarity Comparison **125** can determine similarity of compared elements if their difference is lower than a difference threshold. In further aspects, the rules for determining similarity may include other thresholds. Similarity Comparison **125** enables comparing circumstances including objects with various properties and determining their similarity or match. In one example, a circumstance including an object detected at a distance of 8 m and an angle/bearing of 64° relative to Device **98** may be found similar or matching by Similarity Comparison **125** to a circumstance including the same or similar object detected at a distance of 8.6 m and an angle/bearing of 59° relative to Device **98**. In another example, a circumstance including an object detected as a passenger vehicle may be found similar or matching by Similarity Comparison **125** to a circumstance including an

object detected as a sport utility vehicle. In general, any one or more properties (i.e. existence, type, identity, distance, bearing/angle, location, shape/size, activity, etc.) of one or more objects can be utilized for determining similarity or match of circumstances including objects with various properties. Therefore, Similarity Comparison **125** provides flexibility in comparing and determining similarity of a variety of possible circumstances of Device **98**.

In some embodiments where compared Knowledge Cells **800** include a single Collection of Object Representations **525**, in determining similarity of Knowledge Cells **800**, Similarity Comparison **125** can perform comparison of individual Collections of Object Representations **525** or portions (i.e. Object Representations **625**, Object Properties **630**, etc.) thereof such as comparison of Collection of Object Representations **525** or portions thereof from one Knowledge Cell **800** with Collection of Object Representations **525** or portions thereof from another Knowledge Cell **800**. In some aspects, total equivalence is achieved when Collection of Object Representations **525** or portions thereof from one Knowledge Cell **800** matches Collection of Object Representations **525** or portions thereof from another Knowledge Cell **800**. If total equivalence is not found, Similarity Comparison **125** may attempt to determine substantial or other similarity of compared Knowledge Cells **800**.

In some embodiments, in determining substantial similarity of individually compared Collections of Object Representations **525** (i.e. Collections of Object Representations **525** from the compared Knowledge Cells **800**, etc.), Similarity Comparison **125** can compare one or more Object Representations **625** or portions (i.e. Object Properties **630**, etc.) thereof from one Collection of Object Representations **525** with one or more Object Representations **625** or portions thereof from another Collection of Object Representations **525**. In some aspects, total equivalence is found when all Object Representations **625** or portions thereof from one Collection of Object Representations **525** match all Object Representations **625** or portions thereof from another Collection of Object Representations **525**. In other aspects, if total equivalence is not found, Similarity Comparison **125** may attempt to determine substantial similarity of compared Collections of Object Representations **525**. In one example, substantial similarity can be achieved when most of the Object Representations **625** or portions thereof from the compared Collections of Object Representations **525** match or substantially match. In another example, substantial similarity can be achieved when at least a threshold number (i.e. 1, 2, 4, 7, 18, etc.) or percentage (i.e. 41%, 62%, 79%, 85%, 93%, etc.) of Object Representations **625** or portions thereof from the compared Collections of Object Representations **525** match or substantially match. Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching Object Representations **625** or portions thereof from the compared Collections of Object Representations **525** exceeds a threshold number (i.e. 1, 2, 4, 7, 18, etc.) or a threshold percentage (i.e. 41%, 62%, 79%, 85%, 93%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or percentage of Object Representations **625** or portions thereof from the compared Collections of Object Representations **525** match or substantially match. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In some aspects, Similarity Comparison **125** can utilize the importance (i.e. as indicated by importance index [later described], etc.) of Object Representations **625** or portions thereof for determining substantial similarity of Collections of Object Representations **525**. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to more important Object Representations **625** or portions thereof such as Object Representations **625** representing near Objects **615**, Object Representations **625** representing large Objects **615**, etc., thereby tolerating mismatches in less important Object Representations **625** or portions thereof such as Object Representations **625** representing distant Objects **615**, Object Representations **625** representing small Objects **615**, etc. In general, any Object Representation **625** or portion thereof can be assigned higher or lower importance. In further aspects, Similarity Comparison **125** can omit some of the Object Representations **625** or portions thereof from the comparison in determining substantial similarity of Collections of Object Representations **525**. In one example, Object Representations **625** representing distant Objects **615** can be omitted from comparison. In another example, Object Representations **625** representing small Objects **615** can be omitted from comparison. In general, any Object Representation **625** or portion thereof can be omitted from comparison depending on implementation.

Similarity Comparison **125** can automatically adjust (i.e. increase or decrease) the strictness of the rules for determining substantial similarity of Collections of Object Representations **525**. In some aspects, such adjustment in strictness can be done by Similarity Comparison **125** in response to determining that total equivalence of compared Collections of Object Representations **525** had not been found. Similarity Comparison **125** can keep adjusting the strictness rules until a substantial similarity is found. All the rules or settings of substantial similarity can be set, reset, or adjusted by Similarity Comparison **125** in response to another strictness level determination. For example, Similarity Comparison **125** may attempt to find a match or substantial match in a certain percentage (i.e. 81%, etc.) of Object Representations **625** or portions thereof from the compared Collections of Object Representations **525**. If the comparison does not determine substantial similarity of compared Collections of Object Representations **525**, Similarity Comparison **125** may decide to decrease the strictness of the rules. In response, Similarity Comparison **125** may attempt to find fewer matching or substantially matching Object Representations **625** or portions thereof than in the previous attempt using stricter rules. If the comparison still does not determine substantial similarity of compared Collections of Object Representations **525**, Similarity Comparison **125** may further decrease the strictness (i.e. down to a certain minimum strictness or threshold, etc.) by requiring fewer Object Representations **625** or portions thereof to match or substantially match, thereby further increasing a chance of finding substantial similarity in compared Collections of Object Representations **525**.

Where a reference to Object Representation **625** is used herein it should be understood that a portion of Object Representation **625** (i.e. Object Property **630**, etc.) or a plurality of Object Representations **625** can be used instead of or in addition to the Object Representation **625**. In one example, instead of or in addition to Object Representation **625**, Object Properties **630** and/or other portions that constitute an Object Representation **625** can be compared. In another example, instead of or in addition to Object Representation **625**, plurality of Object Representations **625** can be compared. As such, any operations, rules, logic, and/or functions operating on Object Representation **625** may similarly apply to any portion of Object Representation **625**

and/or a plurality of Object Representations **625** as applicable. In general, whole Object Representations **625**, portions of Object Representations **625**, and/or pluralities of Object Representations **625**, including any operations thereon, can be combined to arrive at desired results. Some or all of the above-described rules, logic, and/or techniques can be utilized alone or in combination with each other or with other rules, logic, and/or techniques. One of ordinary skill in art will recognize that other techniques known in art for determining similarity of Object Representations **625** and/or other data that would be too voluminous to describe are within the scope of this disclosure.

In some embodiments, in determining substantial similarity of Object Representations **625** (i.e. Object Representations **625** from the compared Collections of Object Representations **525**, etc.), Similarity Comparison **125** can compare Object Properties **630** or portions (i.e. characters, words, numbers, etc.) thereof from one Object Representation **625** with Object Properties **630** or portions thereof from another Object Representation **625**. In some aspects, total equivalence is found when all Object Properties **630** or portions thereof of one Object Representation **625** match all Object Properties **630** or portions thereof of another Object Representation **625**. In other aspects, if total equivalence is not found, Similarity Comparison **125** may attempt to determine substantial similarity of compared Object Representations **625**. In one example, substantial similarity can be achieved when most of the Object Properties **630** or portions thereof from the compared Object Representations **625** match or substantially match. In another example, substantial similarity can be achieved when at least a threshold number (i.e. 1, 2, 3, 6, 11, etc.) or percentage (i.e. 55%, 61%, 78%, 82%, 99%, etc.) of Object Properties **630** or portions thereof from the compared Object Representations **625** match or substantially match. Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching Object Properties **630** or portions thereof from the compared Object Representations **625** exceeds a threshold number (i.e. 1, 2, 3, 6, 11, etc.) or a threshold percentage (i.e. 55%, 61%, 78%, 82%, 99%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or percentage of Object Properties **630** or portions thereof from the compared Object Representations **625** match or substantially match. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In further aspects, Similarity Comparison **125** can utilize Categories **635** associated with Object Properties **630** for determining substantial similarity of Object Representations **625**. In one example, Object Properties **630** or portions thereof from the compared Object Representations **625** in a same Category **635** may be compared. This way, Object Properties **630** or portions thereof can be compared with their own peers. In one instance, Object Properties **630** or portions thereof from the compared Object Representations **625** in Category **635** "Type" may be compared. Any text comparison technique can be utilized in such comparing. In another instance, Object Properties **630** or portions thereof from the compared Object Representations **625** in Category **635** "Distance" or "Bearing" may be compared. Any number comparison technique can be utilized in such comparing. In a further instance, Object Properties **630** or portions thereof from the compared Object Representations **625** in Category **635** "Shape" may be compared. Any model, point cloud, or other computer construct comparison technique can be utilized in such comparing. In further aspects, Similarity Comparison **125** can utilize the importance (i.e. as indicated by importance index [later described], etc.) of Object Properties **630** or portions thereof for determining substantial similarity of Object Representations **625**. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to more important Object Properties **630** or portions thereof such as Object Properties **630** or portions thereof in Categories **635** "Type", "Distance", "Bearing", etc., thereby tolerating mismatches in less important Object Properties **630** or portions thereof such as Object Properties **630** or portions thereof in Categories **635** "Identity", "Shape", etc. In general, any Object Property **630** or portion thereof can be assigned higher or lower importance. In further aspects, Similarity Comparison **125** can omit some of the Object Properties **630** or portions thereof from the comparison in determining substantial similarity of Object Representations **625**. In one example, Object Properties **630** or portions thereof in Category **635** "Identity" can be omitted from comparison. In another example, Object Properties **630** or portions thereof in Category **635** "Shape" can be omitted from comparison. In general, any Object Property **630** or portion thereof can be omitted from comparison depending on implementation.

Similarity Comparison **125** can automatically adjust (i.e. increase or decrease) the strictness of the rules for determining substantial similarity of Object Representations **625**. In some aspects, such adjustment in strictness can be done by Similarity Comparison **125** in response to determining that total equivalence of compared Object Representations **625** had not been found. Similarity Comparison **125** can keep adjusting the strictness rules until a substantial similarity is found. All the rules or settings of substantial similarity can be set, reset, or adjusted by Similarity Comparison **125** in response to another strictness level determination. For example, Similarity Comparison **125** may attempt to find a match or substantial match in a certain percentage (i.e. 87%, etc.) of Object Properties **630** or portions thereof from the compared Object Representations **625**. If the comparison does not determine substantial similarity of compared Object Representations **625**, Similarity Comparison **125** may decide to decrease the strictness of the rules. In response, Similarity Comparison **125** may attempt to find fewer matching or substantially matching Object Properties **630** or portions thereof than in the previous attempt using stricter rules. If the comparison still does not determine substantial similarity of compared Object Representations **625**, Similarity Comparison **125** may further decrease the strictness (i.e. down to a certain minimum strictness or threshold, etc.) by requiring fewer Object Properties **630** or portions thereof to match or substantially match, thereby further increasing a chance of finding substantial similarity in compared Object Representations **625**. In further aspects, an adjustment in strictness can be done by Similarity Comparison **125** in response to determining that multiple substantially similar Object Representations **625** had been found. Similarity Comparison **125** can keep adjusting the strictness of the rules until a best of the substantially similar Object Representations **625** is found. For example, Similarity Comparison **125** may attempt to find a match or substantial match in a certain percentage (i.e. 65%, etc.) of Object Properties **630** or portions thereof from the compared Object Representations **625**. If the comparison determines a number of substantially similar Object Representations **625**, Similarity Comparison **125** may decide to increase the strictness of the rules to decrease the number of substantially similar Object Representations **625**. In response, Similarity

Comparison **125** may attempt to find more matching or substantially matching Object Properties **630** or portions thereof in addition to the earlier found Object Properties **630** or portions thereof to limit the number of substantially similar Object Representations **625**. If the comparison still provides more than one substantially similar Object Representation **625**, Similarity Comparison **125** may further increase the strictness by requiring additional Object Properties **630** or portions thereof to match or substantially match, thereby further narrowing the number of substantially similar Object Representations **625** until a best substantially similar Object Representation **625** is found.

Where a reference to Object Property **630** is used herein it should be understood that a portion of Object Property **630** or a plurality of Object Properties **630** can be used instead of or in addition to the Object Property **630**. In one example, instead of or in addition to Object Property **630**, characters, words, numbers, and/or other portions that constitute an Object Property **630** can be compared. In another example, instead of or in addition to Object Property **630**, a plurality of Object Properties **630** can be compared. As such, any operations, rules, logic, and/or functions operating on Object Property **630** may similarly apply to any portion of Object Property **630** and/or a plurality of Object Properties **630** as applicable. In general, whole Object Properties **630**, portions of Object Properties **630**, and/or pluralities of Object Properties **630**, including any operations thereon, can be combined to arrive at desired results. Some or all of the above-described rules, logic, and/or techniques can be utilized alone or in combination with each other or with other rules, logic, and/or techniques. One of ordinary skill in art will recognize that other techniques known in art for determining similarity of Object Properties **630** and/or other data that would be too voluminous to describe are within the scope of this disclosure.

In some embodiments where compared Knowledge Cells **800** include a stream of Collections of Object Representations **525**, in determining similarity of Knowledge Cells **800**, Similarity Comparison **125** can perform collective comparison of Collections of Object Representations **525** or portions (i.e. Object Representations **625**, Object Properties **630**, etc.) thereof such as comparison of a stream of Collections of Object Representations **525** or portions thereof from one Knowledge Cell **800** with a stream of Collections of Object Representations **525** or portions thereof from another Knowledge Cell **800**. Similarity Comparison **125** of collectively compared Collections of Object Representations **525** or portions thereof may include any features, functionalities, and embodiments of the previously described Similarity Comparison **125** of individually compared Collections of Object Representations **525** or portions thereof. In some aspects, total equivalence is found when all Collections of Object Representations **525** or portions thereof from one Knowledge Cell **800** match all Collections of Object Representations **525** or portions thereof from another Knowledge Cell **800**. If total equivalence is not found, Similarity Comparison **125** may attempt to determine substantial or other similarity of compared Knowledge Cells **800**. In one example, substantial similarity can be achieved when most of the Collections of Object Representations **525** or portions thereof from the compared Knowledge Cells **800** match or substantially match. In another example, substantial similarity can be achieved when at least a threshold number (i.e. 1, 2, 4, 9, 33, 138, etc.) or percentage (i.e. 39%, 58%, 77%, 88%, 94%, etc.) of Collections of Object Representations **525** or portions thereof from the compared Knowledge Cells **800** match or substantially match. Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching Collections of Object Representations **525** or portions thereof from the compared Knowledge Cells **800** exceeds a threshold number (i.e. 1, 2, 4, 9, 33, 138, etc.) or a threshold percentage (i.e. 39%, 58%, 77%, 88%, 94%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or percentage of Collections of Object Representations **525** or portions thereof from the compared Knowledge Cells **800** match or substantially match. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In some aspects, Similarity Comparison **125** can utilize the importance (i.e. as indicated by importance index [later described], etc.) of Collections of Object Representations **525** or portions thereof for determining substantial similarity of Knowledge Cells **800**. In one example, substantial similarity can be achieved when matches or substantial matches are found with respect to more important Collections of Object Representations **525** or portions thereof such as more substantive or larger Collections of Object Representations **525** (i.e. Collections of Object Representations **525** comprising a higher number of Object Representations **625**, etc.) or portions thereof, etc., thereby tolerating mismatches in less important Collections of Object Representations **525** or portions thereof such as less substantive or smaller Collections of Object Representations **525** (i.e. Collections of Object Representations **525** comprising a lower number of Object Representations **625**, etc.) or portions thereof, etc. In general, any Collection of Object Representations **525** or portion thereof can be assigned higher or lower importance. In other aspects, Similarity Comparison **125** can utilize the order of Collections of Object Representations **525** or portions thereof for determining substantial similarity of Knowledge Cells **800**. In one example, substantial similarity can be achieved when matches or substantial matches are found in earlier Collections of Object Representations **525** or portions thereof from the compared Knowledge Cells **800**, thereby tolerating mismatches in later Collections of Object Representations **525** or portions thereof. In another example, substantial similarity can be achieved when matches or substantial matches are found in corresponding (i.e. similarity ordered, temporally related, etc.) Collections of Object Representations **525** or portions thereof from the compared Knowledge Cells **800**. In one instance, a 94th Collection of Object Representations **525** or portions thereof from one Knowledge Cell **800** can be compared with a 94th Collection of Object Representations **525** or portions thereof from another Knowledge Cell **800**. In another instance, a 94th Collection of Object Representations **525** or portions thereof from one Knowledge Cell **800** can be compared with a number of Collections of Object Representations **525** or portions thereof around (i.e. preceding and/or following) a 94th Collection of Object Representations **525** from another Knowledge Cell **800**. This way, flexibility can be implemented in finding a substantially similar Collection of Object Representations **525** or portions thereof if the Collections of Object Representations **525** or portions thereof in the compared Knowledge Cells **800** are not perfectly aligned. In a further instance, Similarity Comparison **125** can utilize Dynamic Time Warping (DTW) and/or other techniques known in art for comparing and/or aligning temporal sequences (i.e. streams of Collections of Object Representations **525** or portions thereof, etc.) that may vary in time or speed. In further aspects, Similarity Comparison

125 can omit some of the Collections of Object Representations **525** or portions thereof from the comparison in determining substantial similarity of Knowledge Cells **800**. In one example, less substantive or smaller Collections of Object Representations **525** or portions thereof can be omitted from comparison. In another example, some or all Collections of Object Representations **525** or portions thereof related to a specific time period can be omitted from comparison. In general, any Collection of Object Representations **525** or portion thereof can be omitted from comparison depending on implementation.

Similarity Comparison **125** can automatically adjust (i.e. increase or decrease) the strictness of the rules for determining substantial similarity of Knowledge Cells **800**. In some aspects, such adjustment in strictness can be done by Similarity Comparison **125** in response to determining that total equivalence of compared Knowledge Cells **800** had not been found. Similarity Comparison **125** can keep adjusting the strictness of the rules until substantial similarity is found. All the rules or settings of substantial similarity can be set, reset, or adjusted by Similarity Comparison **125** in response to another strictness level determination. For example, Similarity Comparison **125** may attempt to find a match or substantial match in a certain percentage (i.e. 92%, etc.) of Collections of Object Representations **525** or portions thereof from the compared Knowledge Cells **800**. If the comparison does not determine substantial similarity of compared Knowledge Cells **800**, Similarity Comparison **125** may decide to decrease the strictness of the rules. In response, Similarity Comparison **125** may attempt to find fewer matching or substantially matching Collections of Object Representations **525** or portions thereof than in the previous attempt using stricter rules. If the comparison still does not determine substantial similarity of compared Knowledge Cells **800**, Similarity Comparison **125** may further decrease (i.e. down to a certain minimum strictness or threshold, etc.) the strictness by requiring fewer Collections of Object Representations **525** or portions thereof to match or substantially match, thereby further increasing a chance of finding substantial similarity in compared Knowledge Cells **800**. In further aspects, an adjustment in strictness can be done by Similarity Comparison **125** in response to determining that multiple substantially similar Knowledge Cells **800** had been found. Similarity Comparison **125** can keep adjusting the strictness of the rules until a best of the substantially similar Knowledge Cells **800** is found. For example, Similarity Comparison **125** may attempt to find a match or substantial match in a certain percentage (i.e. 71%, etc.) of Collections of Object Representations **525** or portions thereof from the compared Knowledge Cells **800**. If the comparison determines a number of substantially similar Knowledge Cells **800**, Similarity Comparison **125** may decide to increase the strictness of the rules to decrease the number of substantially similar Knowledge Cells **800**. In response, Similarity Comparison **125** may attempt to find more matching or substantially matching Collections of Object Representations **525** or portions thereof in addition to the earlier found Collections of Object Representations **525** or portions thereof to limit the number of substantially similar Knowledge Cells **800**. If the comparison still provides more than one substantially similar Knowledge Cell **800**, Similarity Comparison **125** may further increase the strictness by requiring additional Collections of Object Representations **525** or portions thereof to match or substantially match, thereby further narrowing the number of substantially similar Knowledge Cells **800** until a best substantially similar Knowledge Cell **800** is found.

Some or all of the aforementioned rules, logic, and/or techniques for determining substantial similarity of Knowledge Cells **800** can be utilized alone or in combination with each other or with other rules, logic, and/or techniques. One of ordinary skill in art will recognize that other techniques known in art for determining similarity of Knowledge Cells **800** and/or other data that would be too voluminous to describe are within the scope of this disclosure.

In any of the comparisons involving numbers such as, for example, Object Properties **630** including numbers (i.e. distances, bearings/angles, etc.), Similarity Comparison **125** can compare a number from one Object Property **630** with a number from another Object Property **630**. In some aspects, total equivalence is found when the number from one Object Property **630** equals the number from another Object Property **630**. In other aspects, if total equality is not found, Similarity Comparison **125** may attempt to determine substantial similarity of the compared numbers using a tolerance or threshold for determining a match. In some aspects, Similarity Comparison **125** can utilize a threshold for acceptable number difference in determining a match of compared numbers. For example, a threshold for acceptable number difference (i.e. absolute difference, etc.) can be set at 10. Therefore, 130 matches or is sufficiently similar to 135 because the number difference (i.e. 5 in this example) is lower than the threshold for acceptable number difference (i.e. 10 in this example, etc.). Furthermore, 130 does not match or is not sufficiently similar to 143 because the number difference (i.e. 13 in this example) is greater than the threshold for acceptable number difference. Any other threshold for acceptable number difference can be used such as 0.024, 1, 8, 15, 77, 197, 2438, 728322, and/or others. In other aspects, Similarity Comparison **125** can utilize a threshold for acceptable percentage difference in determining a match of compared numbers. For example, a threshold for acceptable percentage difference can be set at 10%. Therefore, 100 matches or is sufficiently similar to 106 because the percentage difference (i.e. 6% in this example) is lower than the threshold for acceptable percentage difference (i.e. 10% in this example). Furthermore, 100 does not match or is not sufficiently similar to 84 because the percentage difference (i.e. 16% in this example) is higher than the threshold for acceptable percentage difference. Any other threshold for acceptable percentage difference can be used such as 0.68%, 1%, 3%, 11%, 33%, 69%, 87%, and/or others. The aforementioned thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. Other techniques known in art for comparing numbers can be utilized herein.

In any of the comparisons involving text such as, for example, Object Properties **630** including text (i.e. types, identities, etc.), Similarity Comparison **125** can compare words, characters, and/or other text from one Object Property **630** with words, characters, and/or other text from another Object Property **630**. In some aspects, total equivalence is found when all words, characters, and/or other text from one Object Property **630** match all words, characters, and/or other text from another Object Property **630**. In other aspects, if total equivalence is not found, Similarity Comparison **125** may attempt to determine substantial similarity of compared Object Properties **630**. In one example, substantial similarity can be achieved when most of the words, characters, and/or other text from the compared Object Properties **630** match or substantially match. In another example, substantial similarity can be achieved when at least

a threshold number (i.e. 1, 2, 3, 4, 7, 11, etc.) or percentage (i.e. 38%, 63%, 77%, 84%, 98%, etc.) of words, characters, and/or other text from the compared Object Properties **630** match or substantially match. Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching words, characters, and/or other text from the compared Object Properties **630** exceeds a threshold number (i.e. 1, 2, 3, 4, 7, 11, etc.) or a threshold percentage (i.e. 48%, 63%, 77%, 84%, 98%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or percentage of words, characters, and/or other text from the compared Object Properties **630** match or substantially match. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In further aspects, Similarity Comparison **125** can utilize the importance (i.e. as indicated by importance index [later described], etc.) of words, characters, and/or other text for determining substantial similarity of Object Properties **630**. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to more important words, characters, and/or other text such as longer words and/or other text, thereby tolerating mismatches in less important words, characters, and/or other text such as shorter words and/or other text. In general, any word, character, and/or other text can be assigned higher or lower importance. In further aspects, Similarity Comparison **125** can utilize the order of words, characters, and/or other text for determining substantial similarity of Object Properties **630**. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to front-most words, characters, and/or other text, thereby tolerating mismatches in later words, characters, and/or other text. In further aspects, Similarity Comparison **125** can utilize semantic conversion to account for variations of words and/or other text. In one example, Object Property **630** may include a word “house”. In addition to searching for the exact word in a compared Object Property **630**, Similarity Comparison **125** can employ semantic conversion and attempt to match “home”, “residence”, “dwelling”, “place”, or other semantically similar variations of the word with a meaning “house”. In another example, Object Property **630** may include a word “buy”. In addition to searching for the exact word in a compared Object Property **630**, Similarity Comparison **125** can employ semantic conversion and attempt to match “buying”, “bought”, or other semantically similar variations of the word with a meaning “buy” in different tenses. Any other grammatical analysis or transformation can be utilized to cover the full scope of word and/or other text variations. In some designs, semantic conversion can be implemented using a thesaurus or dictionary. In another example, semantic conversion can be implemented using a table where each row comprises semantically similar variations of a word and/or other text. In further aspects, Similarity Comparison **125** can utilize a language model for understanding or interpreting the concepts contained in the words and/or other text and compare the concepts instead of or in addition to the words and/or other text. Examples of language models include unigram model, n-gram model, neural network language model, bag of words model, and/or others. Any of the techniques for matching of words can similarly be used for matching of concepts. In further aspects, Similarity Comparison **125** can omit some of the words, characters, and/or other text from the comparison in determining substantial similarity of Object Properties **630**. In one example, rear-most words, characters, and/or other text can be omitted from comparison. In another example, shorter words and/or other text can be omitted from comparison. In general, any word, character, and/or other text can be omitted from comparison depending on implementation. Other techniques known in art for comparing words, characters, and/or other text can be utilized herein.

In some embodiments, Similarity Comparison **125** can compare one or more Extra Info **527** (i.e. time information, location information, computed information, contextual information, and/or other information, etc.) in addition to or instead of comparing Collections of Object Representations **525** or portions thereof in determining substantial similarity of Knowledge Cells **800**. Extra Info **527** can be set to be less, equally, or more important (i.e. as indicated by importance index [later described], etc.) than Collections of Object Representations **525**, Object Representations **625**, Object Properties **630**, and/or other elements in the comparison. Since Extra Info **527** may include any contextual or other information that can be useful in determining similarity of any compared elements, Extra Info **527** can be used to enhance any of the aforementioned similarity determinations as applicable.

In some embodiments, Similarity Comparison **125** can also compare one or more Instruction Sets **526** in addition to or instead of comparing Collections of Object Representations **525** or portions thereof in determining substantial similarity of Knowledge Cells **800**. In some aspects, Similarity Comparison **125** can compare portions of Instruction Sets **526** to determine substantial or other similarity of Instruction Sets **526**. Similar to the above-described thresholds, thresholds for the number or percentage of matching portions of the compared Instruction Sets **526** can be utilized in determining substantial or other similarity of the compared Instruction Sets **526**. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In other aspects, Similarity Comparison **125** can compare text (i.e. characters, words, phrases, etc.), numbers, or other data (i.e. bits, etc.) to determine substantial or other similarity of Instruction Sets **526**. Any other comparison technique can be utilized in comparing Instruction Sets **526** in alternate implementations. Instruction Sets **526** can be set to be less, equally, or more important (i.e. as indicated by importance index [later described], etc.) than Collections of Object Representations **525**, Object Representations **625**, Object Properties **630**, Extra Info **527**, and/or other elements in the comparison.

In some embodiments, an importance index (not shown) or other importance ranking technique can be used in any of the previously described comparisons or other processing involving elements of different importance. Importance index indicates importance of the element to or with which the index is assigned or associated. For example, importance index may indicate importance of a Knowledge Cell **800**, Collection of Object Representations **525**, Object Representation **625**, Object Property **630**, Instruction Set **526**, Extra Info **527**, and/or other element to or with which the index is assigned or associated. In some aspects, importance index on a scale from 0 to 1 can be utilized, although, any other range can also be utilized. Importance index can be stored in or associated with the element to which the index pertains. Importance indexes of various elements can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. In one

example, a higher Importance index can be assigned to more substantive or larger Collections of Object Representations **525** (i.e. Collections of Object Representations **525** comprising a higher number of Object Representations **625**, etc.). In another example, a higher importance index can be assigned to Object Representations **625** representing closer, larger, and/or other Objects **615**. Any importance index can be assigned to or associated with any element described herein depending on implementation. Any importance ranking technique can be utilized as or instead of importance index in alternate embodiments.

In some embodiments, Similarity Comparison **125** may generate a similarity index (not shown) for any of the compared elements. Similarity index indicates how well an element is matched with another element. For example, similarity index indicates how well a Knowledge Cell **800**, Collection of Object Representations **525**, Object Representation **625**, Object Property **630**, Instruction Set **526**, Extra Info **527**, and/or other element is matched with a compared element. In some aspects, similarity index on a scale from 0 to 1 can be utilized, although, any other range can also be utilized. Similarity index can be generated by Similarity Comparison **125** whether substantial or other similarity between the compared elements is achieved or not. In one example, similarity index can be determined for a Knowledge Cell **800** based on a ratio/percentage of matched or substantially matched Collections of Object Representations **525** relative to the number of Collections of Object Representations **525** in the compared Knowledge Cell **800**. Specifically, similarity index of 0.91 is determined if 91% of Collections of Object Representations **525** of one Knowledge Cell **800** match or substantially match Collections of Object Representations **525** of another Knowledge Cell **800**. In some designs, importance (i.e. as indicated by importance index, etc.) of one or more Collections of Object Representations **525** can be included in the calculation of a weighted similarity index. Similar determination of similarity index can be implemented with Collections of Object Representations **525**, Object Representations **625**, Object Properties **630**, Instruction Sets **526**, Extra Info **527**, and/or other elements or portions thereof. Any combination of the aforementioned similarity index determinations or calculations can be utilized in alternate embodiments. Any similarity ranking technique can be utilized to determine or calculate similarity index in alternate embodiments.

Referring to FIG. **21**, an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Neural Network **530***a* comprising shortcut Connections **853** is illustrated. In some designs, Knowledge Cells **800** in one Layer **854** of Neural Network **530***a* can be connected with Knowledge Cells **800** in any Layer **854**, not only in a successive Layer **854**, thereby creating shortcuts (i.e. shortcut Connections **853**, etc.) through Neural Network **530***a*. In some aspects, creating a shortcut Connection **853** can be implemented by performing Similarity Comparisons **125** of a Knowledge Cell **800** from Knowledge Structuring Unit **520** with Knowledge Cells **800** in any Layer **854** when applying (i.e. storing, copying, etc.) the Knowledge Cell **800** from Knowledge Structuring Unit **520** onto Neural Network **530***a*. Once created, shortcut Connections **853** enable a wider variety of Knowledge Cells **800** to be considered when selecting a path through Neural Network **530***a*. In some embodiments, Knowledge Structuring Unit **520** structures or generates Knowledge Cells **800** and the system applies them onto Neural Network **530***a*, thereby implementing learning Device's **98** operation in circumstances including objects with various properties. The system can perform Similarity Comparisons **125** of a Knowledge Cell **800** from Knowledge Structuring Unit **520** with Knowledge Cells **800** in one or more Layers **854** of Neural Network **530***a*. If a substantially similar Knowledge Cell **800** is not found in the one or more Layers **854** of Neural Network **530***a*, the system may insert (i.e. copy, store, etc.) the Knowledge Cell **800** from Knowledge Structuring Unit **520** into a Layer **854** of Neural Network **530***a*, and create a Connection **853** to the inserted Knowledge Cell **800** from a prior Knowledge Cell **800** including assigning an occurrence count to the new Connection **853**, calculating a weight of the new Connection **853**, and updating any other Connections **853** originating from the prior Knowledge Cell **800**. On the other hand, if a substantially similar Knowledge Cell **800** is found in the one or more Layers **854** of Neural Network **530***a*, the system may update occurrence count and weight of Connection **853** to that Knowledge Cell **800** from a prior Knowledge Cell **800**, and update any other Connections **853** originating from the prior Knowledge Cell **800**. Any of the previously described and/or other techniques for comparing, inserting, updating, and/or other operations on Knowledge Cells **800**, Connections **853**, Layers **854**, and/or other elements can similarly be utilized in Neural Network **530***a* that comprises shortcut Connections **853**.

Referring to FIG. **22**, an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Graph **530***b* is illustrated. In some aspects, any Knowledge Cell **800** can be connected with any other Knowledge Cell **800** in Graph **530***b*. In other aspects, any Knowledge Cell **800** can be connected with itself and/or any other Knowledge Cell **800** in Graph **530***b*. In some embodiments, Knowledge Structuring Unit **520** structures or generates Knowledge Cells **800** and the system applies (i.e. store, copy, etc.) them onto Graph **530***b*, thereby implementing learning Device's **98** operation in circumstances including objects with various properties. The system can perform Similarity Comparisons **125** of a Knowledge Cell **800** from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Graph **530***b*. If a substantially similar Knowledge Cell **800** is not found in Graph **530***b*, the system may insert (i.e. copy, store, etc.) the Knowledge Cell **800** from Knowledge Structuring Unit **520** into Graph **530***b*, and create a Connection **853** to the inserted Knowledge Cell **800** from a prior Knowledge Cell **800** including assigning an occurrence count to the new Connection **853**, calculating a weight of the new Connection **853**, and updating any other Connections **853** originating from the prior Knowledge Cell **800**. On the other hand, if a substantially similar Knowledge Cell **800** is found in Graph **530***b*, the system may update occurrence count and weight of Connection **853** to that Knowledge Cell **800** from a prior Knowledge Cell **800**, and update any other Connections **853** originating from the prior Knowledge Cell **800**. Any of the previously described and/or other techniques for comparing, inserting, updating, and/or other operations on Knowledge Cells **800**, Connections **853**, and/or other elements can similarly be utilized in Graph **530***b*.

For example, the system can perform Similarity Comparisons **125** of Knowledge Cell **800***ba* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Graph **530***b*. In the case that a substantially similar match is not found, the system may insert Knowledge Cell **800***ha* into Graph **530***b* and copy Knowledge Cell **800***ba* into the inserted Knowledge Cell **800***ha*. The system can then per-

form Similarity Comparisons **125** of Knowledge Cell **800***bb* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Graph **530***b*. In the case that a substantially similar match is found between Knowledge Cell **800***bb* and Knowledge Cell **800***hb*, the system may create Connection **853***h***1** between Knowledge Cell **800***ha* and Knowledge Cell **800***hb* with occurrence count of 1 and weight of 1. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bc* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Graph **530***b*. In the case that a substantially similar match is found between Knowledge Cell **800***bc* and Knowledge Cell **800***hc*, the system may update occurrence count and weight of Connection **853***h***2** between Knowledge Cell **800***hb* and Knowledge Cell **800***hc*, and update weights of other outgoing Connections **853** (one in this example) originating from Knowledge Cell **800***hb* as previously described. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bd* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Graph **530***b*. In the case that a substantially similar match is not found, the system may insert Knowledge Cell **800***hd* into Graph **530***b* and copy Knowledge Cell **800***bd* into the inserted Knowledge Cell **800***hd*. The system may also create Connection **853***h***3** between Knowledge Cell **800***hc* and Knowledge Cell **800***hd* with occurrence count of 1 and weight calculated based on the occurrence count as previously described. The system may also update weights of other outgoing Connections **853** (one in this example) originating from Knowledge Cell **800***hc* as previously described. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***be* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Graph **530***b*. In the case that a substantially similar match is not found, the system may insert Knowledge Cell **800***he* into Graph **530***b* and copy Knowledge Cell **800***be* into the inserted Knowledge Cell **800***he*. The system may also create Connection **853***h***4** between Knowledge Cell **800***hd* and Knowledge Cell **800***he* with occurrence count of 1 and weight of 1. Applying any additional Knowledge Cells **800** from Knowledge Structuring Unit **520** onto Graph **530***b* follows similar logic or process as the above-described.

Referring to FIG. **23**, an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Collection of Sequences **530***c* is illustrated. Collection of Sequences **530***c* comprises the functionality for storing one or more Sequences **533**. Sequence **533** comprises the functionality for storing any number of Knowledge Cells **800**. For example, Knowledge Structuring Unit **520** structures or generates Knowledge Cells **800** and the system applies them onto Collection of Sequences **530***c*, thereby implementing learning Device's **98** operation in circumstances including objects with various properties. The system can perform collective Similarity Comparisons **125** of Knowledge Cells **800** from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Sequences **533** of Collection of Sequences **530***c* to find a Sequence **533** comprising Knowledge Cells **800** that are collectively substantially similar to the Knowledge Cells **800** from Knowledge Structuring Unit **520**. If Sequence **533** comprising such collectively substantially similar Knowledge Cells **800** is not found in Collection of Sequences **530***c*, the system may create a new Sequence **533** comprising the Knowledge Cells **800** from Knowledge Structuring Unit **520** and insert (i.e. copy, store, etc.) the new Sequence **533** into Collection of Sequences **530***c*. On the other hand, if Sequence **533** comprising collectively substantially similar Knowledge Cells **800** is found in Collection of Sequences **530***c*, the system may optionally omit inserting the Knowledge Cells **800** from Knowledge Structuring Unit **520** into Collection of Sequences **530***c* as inserting a similar Sequence **533** may not add much or any additional knowledge. This approach can save storage resources and limit the number of Knowledge Cells **800** that may later need to be processed or compared. In some aspects, a Sequence **533** may include Knowledge Cells **800** relating to a single operation of Device **98**. In other aspects, a Sequence **533** may include Knowledge Cells **800** relating to a part of an operation of Device **98**. In further aspects, one or more long Sequences **533** each including Knowledge Cells **800** of multiple operations of Device **98** can be utilized. In one example, Knowledge Cells **800** of all operations can be stored in a single long Sequence **533** in which case Collection of Sequences **530***c* as a separate element can be omitted. In another example, Knowledge Cells **800** of multiple operations can be included in a plurality of long Sequences **533** such as hourly, daily, weekly, monthly, yearly, or other periodic or other Sequences **533**. Similarity Comparisons **125** can be performed by traversing the one or more long Sequences **533** to find a match or substantially similar match. For instance, the system can perform collective Similarity Comparisons **125** of Knowledge Cells **800** from Knowledge Structuring Unit **520** with Knowledge Cells **800** in subsequences of a long Sequence **533** in incremental or other traversing pattern to find a subsequence comprising Knowledge Cells **800** that are collectively substantially similar to the Knowledge Cells **800** from Knowledge Structuring Unit **520**. The incremental traversing pattern may start from one end of a long Sequence **533** and move the comparison subsequence up or down one or any number of incremental Knowledge Cells **800** at a time. Other traversing patterns or methods can be employed such as starting from the middle of the Sequence **533** and subdividing the resulting sub-sequences in a recursive pattern, or any other traversing pattern or method. If a subsequence comprising collectively substantially similar Knowledge Cells **800** is not found in the long Sequence **533**, the system may concatenate or append the Knowledge Cells **800** from Knowledge Structuring Unit **520** to the long Sequence **533**. In further aspects, Connections **853** can optionally be used in Sequence **533** to connect Knowledge Cells **800**. For example, a Knowledge Cell **800** can be connected not only with a next Knowledge Cell **800** in the Sequence **533**, but also with any other Knowledge Cell **800** in the Sequence **533**, thereby creating alternate routes or shortcuts through the Sequence **533**. Any number of Connections **853** connecting any Knowledge Cells **800** can be utilized. Any of the previously described and/or other techniques for comparing, inserting, updating, and/or other operations on Knowledge Cells **800**, Connections **853**, and/or other elements can similarly be utilized in Sequences **533** and/or Collection of Sequences **530***c*.

In some embodiments, various elements and/or techniques can be utilized in the aforementioned substantial similarity determinations with respect to collectively compared Knowledge Cells **800** and/or other elements. In some aspects, substantial similarity of collectively compared Knowledge Cells **800** can be determined based on similarities or similarity indexes of the individually compared Knowledge Cells **800**. In one example, an average of similarities or similarity indexes of individually compared Knowledge Cells **800** can be used to determine similarity of collectively compared Knowledge Cells **800**. In another example, a weighted average of similarities or similarity

indexes of individually compared Knowledge Cells **800** can be used to determine similarity of collectively compared Knowledge Cells **800**. For instance, to affect the weighting of collective similarity, a higher weight or importance (i.e. importance index, etc.) can be assigned to the similarities or similarity indexes of some Knowledge Cells **800** and lower for other Knowledge Cells **800**. Any higher or lower weight or importance assignment can be implemented. In other aspects, any of the previously described or other thresholds for substantial similarity of individually compared elements can similarly be utilized for collectively compared elements. In one example, substantial similarity of collectively compared Knowledge Cells **800** can be achieved when their collective similarity or similarity index exceeds a similarity threshold. In another example, substantial similarity of collectively compared Knowledge Cells **800** can be achieved when at least a threshold number or percentage of Knowledge Cells **800** from the collectively compared Knowledge Cells **800** match or substantially match. Similarly, substantial similarity of collectively compared Knowledge Cells **800** can be achieved when a number or percentage of matching or substantially matching Knowledge Cells **800** from the collectively compared Knowledge Cells **800** exceeds a threshold. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. Similar elements and/or techniques as the aforementioned can be used for similarity determinations of other collectively compared elements such as Collections of Object Representations **525**, Object Representations **625**, Object Properties **630**, Instruction Sets **526**, Extra Info **527**, and/or others. Similarity determinations of collectively compared elements may include any features, functionalities, and embodiments of Similarity Comparison **125**, and vice versa.

Any of the previously described data structures or arrangements of Knowledge Cells **800** such as Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, and/or others can be used alone, or in combination with each other or with other elements, in alternate embodiments. In one example, a path in Neural Network **530***a* or Graph **530***b* may include its own separate sequence of Knowledge Cells **800** that are not interconnected with Knowledge Cells **800** in other paths. In another example, a part of a path in Neural Network **530***a* or Graph **530***b* may include a sequence of Knowledge Cells **800** interconnected with Knowledge Cells **800** in other paths, whereas, another part of the path may include its own separate sequence of Knowledge Cells **800** that are not interconnected with Knowledge Cells **800** in other paths. Any other combinations or arrangements of Knowledge Cells **800** can be implemented.

Referring to FIG. **24**, an embodiment of determining anticipatory Instruction Sets **526** from a single Knowledge Cell **800** is illustrated. Knowledge Cell **800** may be part of a Knowledgebase **530** (i.e. Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.) such as Collection of Knowledge Cells **530***d*. Decision-making Unit **540** comprises the functionality for anticipating or determining a device's operation in circumstances including objects with various properties. Decision-making Unit **540** comprises the functionality for anticipating or determining Instruction Sets **526** to be used or executed in Device's **98** autonomous operation. In some aspects, Instruction Sets **526** anticipated or determined to be used or executed in Device's **98** autonomous operation may be referred to as anticipatory Instruction Sets **526**, alternate Instruction Sets **526**, and/or other suitable name or reference. Therefore, these terms can be used interchangeably herein depending on context. Decision-making Unit **540** also comprises other disclosed functionalities.

In some aspects, Decision-making Unit **540** may anticipate or determine Instruction Sets **526** (i.e. anticipatory Instruction Sets **526**, etc.) for autonomous Device **98** operation by performing Similarity Comparisons **125** of incoming Collections of Object Representations **525** or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Knowledgebase **530** (i.e. Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.). A Knowledge Cell **800** includes knowledge (i.e. one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**, etc.) of how Device **98** operated in a circumstance including objects with various properties as previously described. When one or more Collections of Object Representations **525** representing objects with similar properties are received in the future, Decision-making Unit **540** can anticipate the Instruction Sets **526** (i.e. anticipatory Instruction Sets **526**, etc.) previously learned in a similar circumstance, thereby enabling autonomous Device **98** operation. In some aspects, Decision-making Unit **540** can perform Similarity Comparisons **125** of incoming Collections of Object Representations **525** from Object Processing Unit **93** with Collections of Object Representations **525** from Knowledge Cells **800** in Knowledgebase **530** (i.e. Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.). If one or more substantially similar Collections of Object Representations **525** or portions thereof are found in a Knowledge Cell **800** from Knowledgebase **530**, Instruction Sets **526** (i.e. anticipatory Instruction Sets **526**, etc.) for autonomous Device **98** operation can be anticipated in Instruction Sets **526** correlated with the one or more Collections of Object Representations **525** from the Knowledge Cell **800**. In some designs, subsequent one or more Instruction Sets **526** for autonomous Device **98** operation can be anticipated in Instruction Sets **526** correlated with subsequent Collections of Object Representations **525** from the Knowledge Cell **800** or other Knowledge Cells **800**, thereby anticipating not only current, but also additional future Instruction Sets **526**. Although, Extra Info **527** is not shown in this and/or other figures for clarity of illustration, it should be noted that any Collection of Object Representations **525**, Instruction Set **526**, and/or other element may include or be associated with Extra Info **527** and that Decision-making Unit **540** can utilize Extra Info **527** for enhanced decision making.

For example, Decision-making Unit **540** can perform Similarity Comparison **125** of Collection of Object Representations **525/1** or portions thereof from Object Processing Unit **93** with Collection of Object Representations **525***a***1** or portions thereof from Knowledge Cell **800***oa*. Collection of Object Representations **525***a***1** or portions thereof from Knowledge Cell **800***oa* may be found substantially similar. Decision-making Unit **540** can anticipate Instruction Sets **526***a***1**-**526***a***3** correlated with Collection of Object Representations **525***a***1**, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparison **125** of Collection of Object Representations **525/2** or portions thereof from Object Processing Unit **93** with Collection of Object Representations **525***a***2** or portions thereof from Knowledge Cell **800***oa*. Collection of

Object Representations **525*a*2** or portions thereof from Knowledge Cell **800*oa*** may be found substantially similar. Decision-making Unit **540** can anticipate Instruction Set **526*a*4** correlated with Collection of Object Representations **525*a*2**, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparison **125** of Collection of Object Representations **525/3** or portions thereof from Object Processing Unit **93** with Collection of Object Representations **525*a*3** or portions thereof from Knowledge Cell **800*oa***. Collection of Object Representations **525*a*3** or portions thereof from Knowledge Cell **800*oa*** may be found substantially similar. Decision-making Unit **540** may not anticipate any Instruction Sets **526** since none are correlated with Collection of Object Representations **525*a*3**. Decision-making Unit **540** can then perform Similarity Comparison **125** of Collection of Object Representations **525/4** or portions thereof from Object Processing Unit **93** with Collection of Object Representations **525*a*4** or portions thereof from Knowledge Cell **800*oa***. Collection of Object Representations **525*a*4** or portions thereof from Knowledge Cell **800*oa*** may not be found substantially similar. Decision-making Unit **540** can then perform Similarity Comparison **125** of Collection of Object Representations **525/5** or portions thereof from Object Processing Unit **93** with Collection of Object Representations **525*a*5** or portions thereof from Knowledge Cell **800*oa***. Collection of Object Representations **525*a*5** or portions thereof from Knowledge Cell **800*oa*** may not be found substantially similar. Decision-making Unit **540** can implement similar logic or process for any additional Collections of Object Representations **525** from Object Processing Unit **93**, and so on.

It should be understood that any of the described elements and/or techniques in the foregoing example can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells **800** or elements (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof would be affected accordingly. In one example, Extra Info **527** can be included in the Similarity Comparisons **125** as previously described. In another example, as history of incoming Collections of Object Representations **525** becomes available, Decision-making Unit **540** can perform collective Similarity Comparisons **125** of the history of Collections of Object Representations **525** or portions thereof from Object Processing Unit **93** with subsequences of Collections of Object Representations **525** or portions thereof from Knowledge Cell **800**. In a further example, the described comparisons in a single Knowledge Cell **800** may be performed on any number of Knowledge Cells **800** sequentially or in parallel. Parallel processors such as a plurality of Processors **11** or cores thereof can be utilized for such parallel processing. In a further example, various arrangements of Collections of Object Representations **525** and/or other elements in a Knowledge Cell **800** can be utilized as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to FIG. **25**, an embodiment of determining anticipatory Instruction Sets **526** by traversing a single Knowledge Cell **800** is illustrated. Knowledge Cell **800** may be part of a Knowledgebase **530** (i.e. Neural Network **530*a***, Graph **530*b***, Collection of Sequences **530*c***, Sequence **533**, Collection of Knowledge Cells **530*d***, etc.) such as Collection of Knowledge Cells **530*d***. For example, Decision-making Unit **540** can perform Similarity Comparison **125** of Collection of Object Representations **525/1** or portions thereof from Object Processing Unit **93** with Collection of Object Representations **525*a*1** or portions thereof from Knowledge Cell **800*oa***. Collection of Object Representations **525*a*1** or portions thereof from Knowledge Cell **800*oa*** may not be found substantially similar. Decision-making Unit **540** can then perform Similarity Comparison **125** of Collection of Object Representations **525/1** or portions thereof from Object Processing Unit **93** with Collection of Object Representations **525*a*2** or portions thereof from Knowledge Cell **800*oa***. Collection of Object Representations **525*a*2** or portions thereof from Knowledge Cell **800*oa*** may not be found substantially similar. Decision-making Unit **540** can then perform Similarity Comparison **125** of Collection of Object Representations **525/1** or portions thereof from Object Processing Unit **93** with Collection of Object Representations **525*a*3** or portions thereof from Knowledge Cell **800*oa***. Collection of Object Representations **525*a*3** or portions thereof from Knowledge Cell **800*oa*** may be found substantially similar. Decision-making Unit **540** may not anticipate any Instruction Sets **526** since none are correlated with Collection of Object Representations **525*a*3**. Decision-making Unit **540** can then perform Similarity Comparison **125** of Collection of Object Representations **525/2** or portions thereof from Object Processing Unit **93** with Collection of Object Representations **525*a*4** or portions thereof from Knowledge Cell **800*oa***. Collection of Object Representations **525*a*4** or portions thereof from Knowledge Cell **800*oa*** may be found substantially similar. Decision-making Unit **540** can anticipate Instruction Sets **526*a*5-526*a*6** correlated with Collection of Object Representations **525*a*4**, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparison **125** of Collection of Object Representations **525/3** or portions thereof from Object Processing Unit **93** with Collection of Object Representations **525*a*5** or portions thereof from Knowledge Cell **800*oa***. Collection of Object Representations **525*a*5** or portions thereof from Knowledge Cell **800*oa*** may be found substantially similar. Decision-making Unit **540** may not anticipate any Instruction Sets **526** since none are correlated with Collection of Object Representations **525*a*5**. Decision-making Unit **540** can implement similar logic or process for any additional Collections of Object Representations **525** from Object Processing Unit **93**, and so on.

It should be understood that any of the described elements and/or techniques in the foregoing example can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells **800** or elements (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof would be affected accordingly. In one example, Extra Info **527** can be included in the Similarity Comparisons **125** as previously described. In another example, as history of incoming Collections of Object Representations **525** becomes available, Decision-making Unit **540** can perform collective Similarity Comparisons **125** of the history of Collections of Object Representations **525** or portions thereof from Object Processing Unit **93** with subsequences of Collections of Object Representations **525** or portions thereof from Knowledge Cell **800**. In a further example, traversing may be performed in incremental traversing pattern such as starting from one end of Knowledge Cell **800** and moving the comparison subsequence up or down the list one or any number of incremental Collections

of Object Representations **525** at a time. Other traversing patterns or methods can be employed such as starting from the middle of the Knowledge Cell **800** and subdividing the resulting subsequence in a recursive pattern, or any other traversing pattern or method. In a further example, the described traversing of a single Knowledge Cell **800** may be performed on any number of Knowledge Cells **800** sequentially or in parallel. Parallel processors such as a plurality of Processors **11** or cores thereof can be utilized for such parallel processing. In a further example, various arrangements of Collections of Object Representations **525** and/or other elements in a Knowledge Cell **800** can be utilized as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to FIG. **26**, an embodiment of determining anticipatory Instruction Sets **526** using collective similarity comparisons is illustrated. For example, Decision-making Unit **540** can perform Similarity Comparisons **125** of Collection of Object Representations **525/1** or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. Collection of Object Representations **525***c***1** or portions thereof from Knowledge Cell **800***rc* may be found substantially similar with highest similarity. Decision-making Unit **540** can anticipate any Instruction Sets **526** (not shown) correlated with Collection of Object Representations **525***c***1**, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform collective Similarity Comparisons **125** of Collections of Object Representations **525/1-525/2** or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. Collections of Object Representations **525***c***1-525***c***2** or portions thereof from Knowledge Cell **800***rc* may be found substantially similar with highest similarity. Decision-making Unit **540** can anticipate any Instruction Sets **526** (not shown) correlated with Collection of Object Representations **525***c***2**, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform collective Similarity Comparisons **125** of Collections of Object Representations **525/1-525/3** or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. Collections of Object Representations **525***d***1-525***d***3** or portions thereof from Knowledge Cell **800***rd* may be found substantially similar with highest similarity. Decision-making Unit **540** can anticipate any Instruction Sets **526** (not shown) correlated with Collection of Object Representations **525***d***3**, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform collective Similarity Comparisons **125** of Collections of Object Representations **525/1-525/4** or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. Collections of Object Representations **525***d***1-525***d***4** or portions thereof from Knowledge Cell **800***rd* may be found substantially similar with highest similarity. Decision-making Unit **540** can anticipate any Instruction Sets **526** (not shown) correlated with Collection of Object Representations **525***d***4**, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform collective Similarity Comparisons **125** of Collections of Object Representations **525/1-525/5** or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. Collections of Object Representations **525***d***1-525***d***5** or portions thereof from Knowledge Cell **800***rd* may be found substantially similar with highest similarity. Decision-making Unit **540** can anticipate any Instruction Sets **526** (not shown) correlated with Collection of Object Representations **525***d***5**, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can implement similar logic or process for any additional Collections of Object Representations **525** from Object Processing Unit **93**, and so on.

In some embodiments, various elements and/or techniques can be utilized in the aforementioned similarity determinations with respect to collectively compared Collections of Object Representations **525** and/or other elements. In some aspects, similarity of collectively compared Collections of Object Representations **525** can be determined based on similarities or similarity indexes of the individually compared Collections of Object Representations **525**. In one example, an average of similarities or similarity indexes of individually compared Collections of Object Representations **525** can be used to determine similarity of collectively compared Collections of Object Representations **525**. In another example, a weighted average of similarities or similarity indexes of individually compared Collections of Object Representations **525** can be used to determine similarity of collectively compared Collections of Object Representations **525**. For instance, to affect the weighting of collective similarity, a higher weight or importance (i.e. importance index, etc.) can be assigned to the similarities or similarity indexes of some (i.e. more substantive or larger, etc.) Collections of Object Representations **525** and lower for other (i.e. less substantive or smaller, etc.) Collections of Object Representations **525**. Any other higher or lower weight or importance assignment can be implemented. In other aspects, any of the previously described or other thresholds for substantial similarity of individually compared elements can be similarly utilized for collectively compared elements. In one example, substantial similarity of collectively compared Collections of Object Representations **525** can be achieved when their collective similarity or similarity index exceeds a similarity threshold. In another example, substantial similarity of collectively compared Collections of Object Representations **525** can be achieved when at least a threshold number or percentage of Collections of Object Representations **525** or portions thereof from the collectively compared Collections of Object Representations **525** match or substantially match. Similarly, substantial similarity of collectively compared Collections of Object Representations **525** can be achieved when a number or percentage of matching or substantially matching Collections of Object Representations **525** or portions thereof from the collectively compared Collections of Object Representations **525** exceeds a threshold. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. Similar elements and/or techniques as the aforementioned can be used for similarity determinations of other collectively compared elements such as Object Representations **625**, Object Properties **630**, Instruction Sets **526**, Extra Info **527**, Knowledge Cells **800**, and/or others. Similarity determinations of collectively compared elements may

include any features, functionalities, and embodiments of Similarity Comparison **125**, and vice versa.

It should be understood that any of the described elements and/or techniques in the foregoing example can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells **800** or elements (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info **527** in Similarity Comparisons **125**, traversing of Knowledge Cells **800** or other elements, using history of Collections of Object Representations **525** or Knowledge Cells **800** for collective Similarity Comparisons **125**, using various arrangements of Collections of Object Representations **525** and/or other elements in a Knowledge Cell **800**, and/or others can similarly be utilized in this example. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to FIG. **27**, an embodiment of determining anticipatory Instruction Sets **526** using Neural Network **530***a* is illustrated. In some aspects, determining anticipatory Instruction Sets **526** using Neural Network **530***a* may include selecting a path of Knowledge Cells **800** or elements (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof through Neural Network **530***a*. Decision-making Unit **540** can utilize various elements and/or techniques for selecting a path through Neural Network **530***a*. Although, these elements and/or techniques are described with respect to Neural Network **530***a* below, they can similarly be used in any Knowledgebase **530** (i.e. Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.) as applicable.

In some embodiments, Decision-making Unit **540** can utilize similarity index in selecting Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof in a path through Neural Network **530***a*. For instance, similarity index may indicate how well one Knowledge Cell **800** or portions thereof are matched with another Knowledge Cell **800** or portions thereof as previously described. In one example, Decision-making Unit **540** may select a Knowledge Cell **800** comprising one or more Collections of Object Representations **525** with highest similarity index even if Connection **853** pointing to that Knowledge Cell **800** has less than the highest weight. Therefore, similarity index or other such element or parameter can override or disregard the weight of a Connection **853** or other element. In another example, Decision-making Unit **540** may select a Knowledge Cell **800** comprising one or more Collections of Object Representations **525** whose similarity index is higher than or equal to a weight of Connection **853** pointing to that Knowledge Cell **800**. In a further example, Decision-making Unit **540** may select a Knowledge Cell **800** comprising one or more Collections of Object Representations **525** whose similarity index is lower than or equal to a weight of Connection **853** pointing to that Knowledge Cell **800**. Similarity index can be set to be more, less, or equally important than a weight of a Connection **853**.

In some embodiments, Decision-making Unit **540** can utilize Connections **853** in selecting Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof in a path through Neural Network **530***a*. In some aspects, Decision-making Unit **540** can take into account weights of Connections **853** among the interconnected Knowledge Cells **800** in choosing from which Knowledge Cell **800** to compare one or more Collections of Object Representations **525** first, second, third, and so on. Specifically, for instance, Decision-making Unit **540** can perform Similarity Comparisons **125** with one or more Collections of Object Representations **525** from Knowledge Cell **800** pointed to by the highest weight Connection **853** first, Collections of Object Representations **525** from Knowledge Cell **800** pointed to by the second highest weight Connection **853** second, and so on. In other aspects, Decision-making Unit **540** can stop performing Similarity Comparisons **125** as soon as it finds one or more substantially similar Collections of Object Representations **525** in an interconnected Knowledge Cell **800**. In further aspects, Decision-making Unit **540** may only follow the highest weight Connection **853** to arrive at a Knowledge Cell **800** comprising one or more Collections of Object Representations **525** to be compared, thereby disregarding Connections **853** with less than the highest weight. In further aspects, Decision-making Unit **540** may ignore weights and/or other parameters of Connections **853**. In further aspects, Decision-making Unit **540** may ignore Connections **853**.

In some embodiments, Decision-making Unit **540** can utilize a bias to adjust similarity index, weight of a Connection **853**, and/or other element or parameter used in selecting Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof in a path through Neural Network **530***a*. In one example, Decision-making Unit **540** may select a Knowledge Cell **800** comprising one or more Collections of Object Representations **525** whose similarity index multiplied by or adjusted for a bias is higher than or equal to a weight of Connection **853** pointing to that Knowledge Cell **800**. In another example, Decision-making Unit **540** may select a Knowledge Cell **800** comprising one or more Collections of Object Representations **525** whose similarity index multiplied by or adjusted for a bias is lower than or equal to a weight of Connection **853** pointing to that Knowledge Cell **800**. In a further example, bias can be used to resolve deadlock situations where similarity index is equal to a weight of a Connection **853**. In some aspects, bias can be expressed in percentages such as 0.3 percent, 1.2 percent, 25.7 percent, 79.8 percent, 99.9 percent, 100.1 percent, 155.4 percent, 298.6 percent, 1105.5 percent, and so on. For example, a bias below 100 percent decreases an element or parameter to which it is applied, a bias equal to 100 percent does not change the element or parameter to which it is applied, and a bias higher than 100 percent increases the element or parameter to which it is applied. In general, any amount of bias can be utilized depending on implementation. Bias can be applied to one or more of a weight of a Connection **853**, similarity index, any other element or parameter, and/or all or any combination of them. Also, different biases can be applied to each of a weight of a Connection **853**, similarity index, or any other element or parameter. For example, 30 percent bias can be applied to similarity index and 15 percent bias can be applied to a weight of a Connection **853**. Also, different biases can be applied to various Layers **854** of Neural Network **530***a*, and/or other disclosed elements. Bias can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input.

Any other element and/or technique can be utilized in selecting Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof in a path through Neural Network **530***a*.

In some embodiments, Neural Network **530***a* may include knowledge (i.e. interconnected Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**, etc.) of how Device **98** operated in circumstances including objects with various properties. In some aspects, determining anticipatory Instruction Sets **526** using Neural Network **530***a* may include selecting a path of Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof through Neural Network **530***a*. Individual and/or collective Similarity Comparisons **125** can be used to determine substantial similarity of the individually and/or collectively compared Collections of Object Representations **525** or portions thereof. Substantial similarity may be used primarily for selecting a path through Neural Network **530***a*, whereas, weight of any Connection **853** may be used secondarily or not at all.

For example, Decision-making Unit **540** can perform Similarity Comparisons **125** of Collections of Object Representations **525***a***1**-**525***an* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from one or more Knowledge Cells **800** in Layer **854***a* (or any other one or more Layers **854**, etc.). Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***ta* may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525***b***1**-**525***bn* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from one or more Knowledge Cells **800** in Layer **854***b* interconnected with Knowledge Cell **800***ta*. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***tb* may be found collectively substantially similar with highest similarity, thus, Decision-making Unit **540** may follow Connection **853***t***1** disregarding its less than highest weight. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Since Connection **853***t***2** is the only connection from Knowledge Cell **800***tb*, Decision-making Unit **540** may follow Connection **853***t***2** and perform Similarity Comparisons **125** of Collections of Object Representations **525***c***1**-**525***cn* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***tc* in Layer **854***c*. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***tc* may be found collectively substantially similar. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525***d***1**-**525***dn* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from one or more Knowledge Cells **800** in Layer **854***d* interconnected with Knowledge Cell **800***tc*. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***td* may be found collectively substantially similar with highest similarity, thus, Decision-making Unit **540** may follow Connection **853***t***3**. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525***e***1**-**525***en* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from one or more Knowledge Cells **800** in Layer **854***e* interconnected with Knowledge Cell **800***td*. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***te* may be found collectively substantially similar with highest similarity, thus, Decision-making Unit **540** may follow Connection **853***t***4**. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can implement similar logic or process for any additional Collections of Object Representations **525** from Object Processing Unit **93**, and so on.

The foregoing exemplary embodiment provides an example of utilizing a combination of collective Similarity Comparisons **125**, individual Similarity Comparisons **125**, Connections **853**, and/or other elements or techniques. It should be understood that any of these elements and/or techniques can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof in a path through Neural Network **530***a* would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info **527** in Similarity Comparisons **125**, traversing of Knowledge Cells **800** or other elements, using history of Collections of Object Representations **525** or Knowledge Cells **800** for collective Similarity Comparisons **125**, using various arrangements of Collections of Object Representations **525** and/or other elements in a Knowledge Cell **800**, and/or others can similarly be utilized in this example. These elements and/or techniques can similarly be utilized in Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, and/or other data structures or arrangements. In some aspects, instead of anticipating Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525**, Decision-making Unit **540** can anticipate instruction Sets **526** correlated with substantially similar streams of Collections of Object Representations **525**. In other aspects, any time that substantial similarity or other similarity thresh-

old is not achieved in compared Collections of Object Representations **525** or portions thereof from any of the Knowledge Cells **800**, Decision-making Unit **540** can decide to look for a substantially or otherwise similar Collections of Object Representations **525** or portions thereof in Knowledge Cells **800** elsewhere in Neural Network **530***a* such as in any Layer **854** subsequent to a current Layer **854**, in the first Layer **854**, in the entire Neural Network **530***a*, and/or others, even if such Knowledge Cell **800** may be unconnected with a prior Knowledge Cell **800**. It should be noted that any of Collections of Object Representations **525***a***1**-**525***an*, Collections of Object Representations **525***b***1**-**525***bn*, Collections of Object Representations **525***c***1**-**525***cn*, Collections of Object Representations **525***d***1**-**525***dn*, Collections of Object Representations **525***e***1**-**525***en*, etc. may include one Collection of Object Representations **525** or a stream of Collections of Object Representations **525**. It should also be noted that any Knowledge Cell **800** may include one Collection of Object Representations **525** or a stream of Collections of Object Representations **525** as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to FIG. **28**, an embodiment of determining anticipatory Instruction Sets **526** using Graph **530***b* is illustrated. Graph **530***b* may include knowledge (i.e. interconnected Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**, etc.) of how Device **98** operated in circumstances including objects with various properties. In some aspects, determining anticipatory Instruction Sets **526** using Graph **530***b* may include selecting a path of Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof through Graph **530***b*. Individual and/or collective Similarity Comparisons **125** can be used to determine substantial similarity of the individually and/or collectively compared Collections of Object Representations **525** or portions thereof. Substantial similarity may be used primarily for selecting a path through Graph **530***b*, whereas, weight of any Connection **853** may be used secondarily or not at all.

For example, Decision-making Unit **540** can perform Similarity Comparisons **125** of Collections of Object Representations **525***a***1**-**525***an* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from one or more Knowledge Cells **800** in Graph **530***b*. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***ua* may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525***b***1**-**525***bn* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from one or more Knowledge Cells **800** in Graph **530***b* interconnected with Knowledge Cell **800***ua* by outgoing Connections **853**. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***ub* may be found collectively substantially similar with highest similarity, thus, Decision-making Unit **540** may follow Connection **853***u***1** disregarding its less than highest weight. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525***c***1**-**525***cn* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from one or more Knowledge Cells **800** in Graph **530***b* interconnected with Knowledge Cell **800***ub* by outgoing Connections **853**. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***uc* may be found collectively substantially similar with highest similarity, thus, Decision-making Unit **540** may follow Connection **853***u***2** disregarding its less than highest weight. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Since Connection **853***u***3** is the only connection from Knowledge Cell **800***uc*, Decision-making Unit **540** may follow Connection **853***u***3** and perform Similarity Comparisons **125** of Collections of Object Representations **525***d***1**-**525***dn* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***ud* in Graph **530***b*. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***ud* may be found collectively substantially similar. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525***e***1**-**525***en* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from one or more Knowledge Cells **800** in Graph **530***b* interconnected with Knowledge Cell **800***ud* by outgoing Connections **853**. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***ue* may be found collectively substantially similar with highest similarity, thus, Decision-making Unit **540** may follow Connection **853***u***4**. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can implement similar logic or process for any additional Collections of Object Representations **525** from Object Processing Unit **93**, and so on.

The foregoing exemplary embodiment provides an example of utilizing a combination of collective Similarity Comparisons **125**, individual Similarity Comparisons **125**, Connections **853**, and/or other elements or techniques. It should be understood that any of these elements and/or techniques can be omitted, used in a different combination,

or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof in a path through Graph **530***b* would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info **527** in Similarity Comparisons **125**, traversing of Knowledge Cells **800** or other elements, using history of Collections of Object Representations **525** or Knowledge Cells **800** in collective Similarity Comparisons **125**, using various arrangements of Collections of Object Representations **525** and/or other elements in a Knowledge Cell **800**, and/or others can similarly be utilized in this example. These elements and/or techniques can similarly be utilized in Neural Network **530***a*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, and/or other data structures or arrangements. In some aspects, instead of anticipating Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525**, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially matching streams of Collections of Object Representations **525**. In other aspects, any time that substantial similarity or other similarity threshold is not achieved in compared Collections of Object Representations **525** or portions thereof of any of the Knowledge Cells **800**, Decision-making Unit **540** can decide to look for a substantially or otherwise similar Collections of Object Representations **525** or portions thereof in Knowledge Cells **800** elsewhere in Graph **530***b* even if such Knowledge Cell **800** may be unconnected with a prior Knowledge Cell **800**. It should be noted that any of Collections of Object Representations **525***a***1**-**525***an*, Collections of Object Representations **525***b***1**-**525***bn*, Collections of Object Representations **525***c***1**-**525***cn*, Collections of Object Representations **525***d***1**-**525***dn*, Collections of Object Representations **525***e***1**-**525***en*, etc. may include one Collection of Object Representations **525** or a stream of Collections of Object Representations **525**. It should also be noted that any Knowledge Cell **800** may include one Collection of Object Representations **525** or a stream of Collections of Object Representations **525** as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to FIG. **29**, an embodiment of determining anticipatory Instruction Sets **526** using Collection of Sequences **530***c* is illustrated. Collection of Sequences **530***c* may include knowledge (i.e. sequences of Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**, etc.) of how Device **98** operated in circumstances including objects with various properties. In some aspects, determining anticipatory Instruction Sets **526** for autonomous Device **98** operation using Collection of Sequences **530***c* may include selecting a Sequence **533** of Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof from Collection of Sequences **530***c*. Individual and/or collective Similarity Comparisons **125** can be used to determine substantial similarity of the individually and/or collectively compared Collections of Object Representations **525** or portions thereof.

For example, Decision-making Unit **540** can perform Similarity Comparisons **125** of Collections of Object Representations **525***a***1**-**525***an* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in one or more Sequences **533** of Collection of Sequences **530***c*. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***ca* in Sequence **533***wc* may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525***a***1**-**525***an* and **525***b***1**-**525***bn* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Sequences **533** of Collection of Sequences **530***c*. Collections of Object Representations **525** or portions thereof from Knowledge Cells **800***ca*-**800***cb* in Sequence **533***wc* may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525***a***1**-**525***an*, **525***b***1**-**525***bn*, and **525***c***1**-**525***cn* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Sequences **533** of Collection of Sequences **530***c*. Collections of Object Representations **525** or portions thereof from Knowledge Cells **800***da*-**800***dc* in Sequence **533***wd* may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525***a***1**-**525***an*, **525***b***1**-**525***bn*, **525***c***1**-**525***cn*, and **525***d***1**-**525***dn* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Sequences **533** of Collection of Sequences **530***c*. Collections of Object Representations **525** or portions thereof from Knowledge Cells **800***da*-**800***dd* in Sequence **533***wd* may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525***a***1**-**525***an*, **525***b***1**-**525***bn*, **525***c***1**-**525***cn*, **525***d***1**-**525***dn*, and **525***e***1**-**525***en* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Sequences **533** of Collection of Sequences **530***c*. Collections of Object Representations

525 or portions thereof from Knowledge Cells **800***da*-**800***de* in Sequence **533***wd* may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can implement similar logic or process for any additional Collections of Object Representations **525** from Object Processing Unit **93**, and so on.

The foregoing exemplary embodiment provides an example of utilizing a combination of collective Similarity Comparisons **125**, individual Similarity Comparisons **125**, and/or other elements or techniques. It should be understood that any of these elements and/or techniques can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Sequence **533** of Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info **527** in Similarity Comparisons **125**, traversing of Knowledge Cells **800** or other elements, using history of Collections of Object Representations **525** or Knowledge Cells **800** in collective Similarity Comparisons **125**, using various arrangements of Collections of Object Representations **525** and/or other elements in a Knowledge Cell **800**, and/or others can similarly be utilized in this example. These elements and/or techniques can similarly be utilized in Neural Network **530***a*, Graph **530***b*, Collection of Knowledge Cells **530***d*, and/or other data structures or arrangements. In some aspects, instead of anticipating Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525**, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially matching streams of Collections of Object Representations **525**. In other aspects, any time that substantial similarity or other similarity threshold is not achieved in compared Collections of Object Representations **525** or portions thereof from any of the Knowledge Cells **800**, Decision-making Unit **540** can decide to look for a substantially or otherwise similar Collections of Object Representations **525** or portions thereof in Knowledge Cells **800** elsewhere in Collection of Sequences **530***c* such as in different Sequences **533**. It should be noted that any of Collections of Object Representations **525***a***1**-**525***an*, Collections of Object Representations **525***b***1**-**525***bn*, Collections of Object Representations **525***c***1**-**525***cn*, Collections of Object Representations **525***d***1**-**525***dn*, Collections of Object Representations **525***e***1**-**525***en*, etc. may include one Collection of Object Representations **525** or a stream of Collections of Object Representations **525**. It should also be noted that any Knowledge Cell **800** may include one Collection of Object Representations **525** or a stream of Collections of Object Representations **525** as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring now to Modification Interface **130**. Modification Interface **130** comprises the functionality for modifying execution and/or functionality of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. Modification Interface **130** comprises the functionality for modifying execution and/or functionality of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element at runtime. Modification Interface **130** comprises the functionality for modifying execution and/or functionality of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element based on anticipatory Instruction Sets **526**. In one example, Modification Interface **130** comprises the functionality to access, modify, and/or perform other manipulations on runtime engine/environment, virtual machine, operating system, compiler, just-in-time (JIT) compiler, interpreter, translator, execution stack, file, object, data structure, and/or other computing system elements. In another example, Modification Interface **130** comprises the functionality to access, modify, and/or perform other manipulations on memory, storage, bus, interfaces, and/or other computing system elements. In a further example, Modification Interface **130** comprises the functionality to access, modify, and/or perform other manipulations on Processor **11** registers and/or other Processor **11** elements. In a further example, Modification Interface **130** comprises the functionality to access, modify, and/or perform other manipulations on inputs and/or outputs of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. In a further example, Modification Interface **130** comprises the functionality to access, create, delete, modify, and/or perform other manipulations on functions, methods, procedures, routines, subroutines, and/or other elements of Application Program **18**. In a further example, Modification Interface **130** comprises the functionality to access, create, delete, modify, and/or perform other manipulations on source code, bytecode, compiled, interpreted, or otherwise translated code, machine code, and/or other code. In a further example, Modification Interface **130** comprises the functionality to access, create, delete, modify, and/or perform other manipulations on values, variables, parameters, and/or other data or information. Modification Interface **130** comprises any features, functionalities, and embodiments of Acquisition Interface **120**, and vice versa. Modification Interface **130** also comprises other disclosed functionalities.

Modification Interface **130** can employ various techniques for modifying execution and/or functionality of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. In some aspects, some of the previously described techniques and/or tools can be utilized. Code instrumentation, for instance, may involve inserting additional code, overwriting or rewriting existing code, and/or branching to a separate segment of code in Application Program **18** as previously described. For example, instrumented code may include the following:

```
Device1.moveLeft(23);
modifyApplication( );
```

In the above sample code, instrumented call to Modification Interface's **130** function (i.e. modifyApplication( ), etc.) can be placed after a function (i.e. Device1.moveLeft(23), etc.) of Application Program **18**. Similar call to an application modifying function can be placed after or before some or all functions/routines/subroutines, some or all lines of code, some or all statements, some or all instructions or instruction sets, some or all basic blocks, and/or some or all other code segments of Application Program **18**. One or more application modifying function calls can be placed anywhere in Application Program's **18** code and can be executed at any points in Application Program's **18** execution. The application modifying function (i.e. modifyApplication( ), etc.) may include Artificial Intelligence Unit **110**-determined anticipa-

tory Instruction Sets **526** that can modify execution and/or functionality of Application Program **18**. In some embodiments, the previously described obtaining Application Program's **18** instruction sets, data, and/or other information as well as modifying execution and/or functionality of Application Program **18** can be implemented in a single function that performs both tasks (i.e. traceAndModifyApplication( ), etc.).

In some embodiments, various computing systems and/or platforms may provide native tools for modifying execution and/or functionality of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. Independent vendors may provide tools with similar functionalities that can be utilized across different platforms. These tools enable a wide range of techniques or capabilities such as instrumentation, self-modifying code capabilities, dynamic code capabilities, branching, code rewriting, code overwriting, hot swapping, accessing and/or modifying objects or data structures, accessing and/or modifying functions/routines/subroutines, accessing and/or modifying variable or parameter values, accessing and/or modifying processor registers, accessing and/or modifying inputs and/or outputs, providing runtime memory access, and/or other capabilities. One of ordinary skill in art will understand that, while all possible variations of the techniques for modifying execution and/or functionality of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element are too voluminous to describe, these techniques are within the scope of this disclosure.

In one example, modifying execution and/or functionality of Application Program **18** can be implemented through utilizing metaprogramming techniques, which include applications that can self-modify or that can create, modify, and/or manipulate other applications. Self-modifying code, dynamic code, reflection, and/or other techniques can be used to facilitate metaprogramming. In some aspects, metaprogramming is facilitated through a programming language's ability to access and manipulate the internals of the runtime engine directly or via an API. In other aspects, metaprogramming is facilitated through dynamic execution of expressions (i.e. anticipatory Instruction Sets **526**, etc.) that can be created and/or executed at runtime. In yet other aspects, metaprogramming is facilitated through application modification tools, which can perform modifications on an application regardless of whether the application's programming language enables metaprogramming capabilities. Some operating systems may protect an application loaded into memory by restricting access to the loaded application. This protection mechanism can be circumvented by utilizing operating system's, processor's, and/or other low level features or commands to unprotect the loaded application. For example, a self-modifying application may modify the in-memory image of itself. To do so, the application can obtain the in-memory address of its code. The application may then change the operating system's or platform's protection on this memory range allowing it to modify the code (i.e. insert anticipatory Instruction Sets **526**, etc.). In addition to a self-modifying application, one application can utilize similar technique to modify another application. Linux mprotect command or similar commends of other operating systems can be used to change protection (i.e. unprotect, etc.) for a region of memory, for example. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program **18** can be implemented through native capabilities of dynamic, interpreted, and/or scripting programming languages and/or platforms. Most of these languages and/or platforms can perform functionalities at runtime that static programming languages may perform during compilation. Dynamic, interpreted, and/or scripting languages provide native functionalities such as self-modification of code, dynamic code, extending the application, adding new code, extending objects and definitions, and/or other functionalities that can modify an application's execution and/or functionality at runtime. Examples of dynamic, interpreted, and/or scripting languages include Lisp, Perl, PHP, JavaScript, Ruby, Python, Smalltalk, Tcl, VBScript, and/or others. Similar functionalities can also be provided in languages such as Java, C, and/or others using reflection. Reflection includes the ability of an application to examine and modify the structure and behavior of the application at runtime. For example, JavaScript can modify its own code as it runs by utilizing Function object constructor as follows:

```
myFunc=new Function(arg1, arg2, argN, functionBody);
```

The sample code above causes a new function object to be created with the specified arguments and body. The body and/or arguments of the new function object may include new instruction sets (i.e. anticipatory Instruction Sets **526**, etc.). The new function can be invoked as any other function in the original code. In another example, JavaScript can utilize eval method that accepts a string of JavaScript statements (i.e. anticipatory Instruction Sets **526**, etc.) and execute them as if they were within the original code. An example of how eval method can be used to modify an application includes the following JavaScript code:

```
anticipatoryInstr='Device1.moveForward(27);';
if (anticipatoryInstr!=" " && anticipatoryInstr!=null)
{
eval(anticipatoryInstr);
}
```

In the sample code above, Artificial Intelligence Unit **110** may generate anticipatory Instruction Set **526** (i.e. 'Device1.moveForward(27)' for moving a Device1 forward 27 units, etc.) and save it in anticipatoryInstr variable, which eval method can then execute. Lisp is another example of dynamic, interpreted, and/or scripting language that includes similar capabilities as previously described JavaScript. For example, Lisp's compile command can create a function at runtime, eval command may parse and evaluate an expression at runtime, and exec command may execute a given instruction set (i.e. string, etc.) at runtime. In another example, dynamic as well as some non-dynamic languages may provide macros, which combine code introspection and/or eval capabilities. In some aspects, macros can access inner workings of the compiler, interpreter, virtual machine, runtime environment/engine, and/or other components of the computing platform enabling the definition of language-like constructs and/or generation of a complete program or sections thereof. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program **18** can be implemented through dynamic code, dynamic class loading, reflection, and/or other native functionalities of a programming language or platform. In static applications or static programming, a class can be defined and/or loaded at compile time. Conversely, in dynamic applications or dynamic programming, a class can be loaded into a running environment at runtime. For example, Java Runtime Environment (JRE) may not require that all classes be loaded at compile time and class loading can occur when a class is first referenced at runtime. Dynamic class loading enables inclusion or

injection of on-demand code and/or functionalities at runtime. System provided or custom class loaders may enable loading of classes into the running environment. Custom class loaders can be created to enable custom functionalities such as, for example, specifying a remote location from which a class can be loaded. In addition to dynamic loading of a pre-defined class, a class can also be created at runtime. In some aspects, a class source code can be created at runtime. A compiler such as javac, com.sun.tools.javac.Main, javax.tools, javax.tools.JavaCompiler, and/or other packages can then be utilized to compile the source code. Javac, com.sun.tools.javac.Main, javax.tools, javax.tools.JavaCompiler, and/or other packages may include an interface to invoke Java compiler from within a running application. A Java compiler may accept source code in a file, string, object (i.e. Java String, StringBuffer, CharSequence, etc.) and/or other source, and may generate Java bytecode (i.e. class file, etc.). Once compiled, a class loader can then load the compiled class into the running environment. In other aspects, a tool such as Javaassist (i.e. Java programming assistant) can be utilized to enable an application to create or modify a class at runtime. Javassist may include a Java library that provides functionalities to create and/or manipulate Java bytecode of an application as well as reflection capabilities. Javassist may provide source-level and bytecode-level APIs. Using the source-level API, a class can be created and/or modified using only source code, which Javassist may compile seamlessly on the fly. Javassist source-level API can therefore be used without knowledge of Java bytecode specification. Bytecode-level API enables creating and/or editing a class bytecode directly. In yet other aspects, similar functionalities to the aforementioned ones may be provided in tools such as Apache Commons BCEL (Byte Code Engineering Library), ObjectWeb ASM, CGLIB (Byte Code Generation Library), and/or others. Once a dynamic code or class is created and loaded, reflection in high-level programming languages such as Java and/or others can be used to manipulate or change the runtime behavior of an application. Examples of reflective programming languages and/or platforms include Java, JavaScript, Smalltalk, Lisp, Python, .NET Common Language Runtime (CLR), Tcl, Ruby, Perl, PHP, Scheme, PL/SQL, and/or others. Reflection can be used in an application to access, examine, modify, and/or manipulate a loaded class and/or its elements. Reflection in Java can be implemented by utilizing a reflection API such as java.lang.Reflect package. The reflection API provides functionalities such as, for example, loading or reloading a class, instantiating a new instance of a class, determining class and instance methods, invoking class and instance methods, accessing and manipulating a class, fields, methods and constructors, determining the modifiers for fields, methods, classes, and interfaces, and/or other functionalities. The above described dynamic code, dynamic class loading, reflection, and/or other functionalities are similarly provided in the .NET platform through its tools such as, for example, System.CodeDom.Compiler namespace, System.Reflection.Emit namespace, and/or other native or other .NET tools. Other platforms in addition to Java and .NET may provide similar tools and/or functionalities. In some designs, dynamic code, dynamic class loading, reflection, and/or other functionalities can be used to facilitate modification of an application by inserting or injecting instruction sets (i.e. anticipatory Instruction Sets **526**, etc.) into a running application. For example, an existing or dynamically created class comprising DCADO Unit **100** functionalities can be loaded into a running application through manual, automatic, or dynamic instrumentation. Once the class is created and loaded, an instance of DCADO Unit **100** class may be constructed. The instance of DCADO Unit **100** can then take or exert control of the application and/or implement alternate instruction sets (i.e. anticipatory Instruction Sets **526**, etc.) at any point in the application's execution. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program **18** can be implemented through independent tools that can be utilized across different platforms. Such tools provide instrumentation and/or other capabilities on more than one platform or computing system and may facilitate application modification or insertion of instruction sets (i.e. anticipatory Instruction Sets **526**, etc.). Examples of these tools include Pin, DynamoRIO, DynInst, Kprobes, KernInst, OpenPAT, DTrace, SystemTap, and/or others. In some aspects, Pin and/or any of its elements, methods, and/or techniques can be utilized for dynamic instrumentation. Pin can perform instrumentation by taking control of an application after it loads into memory. Pin may insert itself into the address space of an executing application enabling it to take control. Pin JIT compiler can then compile and implement alternate code (i.e. anticipatory Instruction Sets **526**, etc.). Pin provides an extensive API for instrumentation at several abstraction levels. Pin supports two modes of instrumentation, JIT mode and probe mode. JIT mode uses a just-in-time compiler to insert instrumentation and recompile program code while probe mode uses code trampolines for instrumentation. Pin was designed for architecture and operating system independence. In other aspects, KernInst and/or any of its elements, methods, and/or techniques can be utilized for dynamic instrumentation. KernInst includes an instrumentation framework designed for dynamically inserting code into a running kernel of an operating system. KernInst implements probe-based dynamic instrumentation where code can be inserted, changed, and/or removed at will. Kerninst API enables client tools to construct their own tools for dynamic kernel instrumentation to suit variety of purposes such as insertion of alternate instruction sets (i.e. anticipatory Instruction Sets **526**, etc.). Client tools can communicate with KernInst over a network (i.e. internet, wireless network, LAW, WAN, etc). Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program **18** can be implemented through utilizing operating system's native tools or capabilities such as Unix ptrace command. Ptrace includes a system call that may enable one process to control another allowing the controller to inspect and manipulate the internal state of its target. Ptrace can be used to modify a running application such as modify an application with alternate instruction sets (i.e. anticipatory Instruction Sets **526**, etc.). By attaching to an application using the ptrace call, the controlling application can gain extensive control over the operation of its target. This may include manipulation of its instruction sets, execution path, file descriptors, memory, registers, and/or other components. Ptrace can single-step through the targets code, observe and intercept system calls and their results, manipulate the targets signal handlers, receive and send signals on the target's behalf, and/or perform other operations within the target application. Ptrace's ability to write into the target application's memory space enables the controller to modify the running code of

the target application. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program **18** can be implemented through utilizing just-in-time (JIT) compiling. JIT compilation (also known as dynamic translation, dynamic compilation, etc.) includes compilation performed during an application's execution (i.e. runtime, etc.). A code can be compiled when it is about to be executed, and it may be cached and reused later without the need for additional compilation. In some aspects, a JIT compiler can convert source code or byte code into machine code. In other aspects, a JIT compiler can convert source code into byte code. JIT compiling may be performed directly in memory. For example, JIT compiler can output machine code directly into memory and immediately execute it. Platforms such as Java, .NET, and/or others may implement JIT compilation as their native functionality. Platform independent tools for custom system design may include JIT compilation functionalities as well. In some aspects, JIT compilation includes redirecting application's execution to a JIT compiler from a specific entry point. For example, Pin can insert its JIT compiler into the address space of an application. Once execution is redirected to it, JIT compiler may receive alternate instruction sets (i.e. anticipatory Instruction Sets **526**, etc.) immediately before their compilation. The JIT compiled instruction sets can be stored in memory or another repository from where they may be retrieved and executed. Alternatively, for example, JIT compiler can create a copy of the original application code or a segment thereof, and insert alternate code (i.e. anticipatory Instruction Sets **526**, etc.) before compiling the modified code copy. In some aspects, JIT compiler may include a specialized memory such as fast cache memory dedicated to JIT compiler functionalities from which the modified code can be fetched rapidly. JIT compilation and/or any compilation in general may include compilation, interpretation, or other translation into machine code, bytecode, and/or other formats or types of code. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program **18** can be implemented through dynamic recompilation. Dynamic recompilation includes recompiling an application or part thereof during execution. An application can be modified with alternate features or instruction sets that may take effect after recompilation. Dynamic recompilation may be practical in various types of applications including object oriented, event driven, forms based, and/or other applications. In a typical windows-based application, most of the action after initial startup occurs in response to user or system events such as moving the mouse, selecting a menu option, typing text, running a scheduled task, making a network connection, and/or other events when an event handler is called to perform an operation appropriate for the event. Generally, when no events are being generated, the application is idle. For example, when an event occurs and an appropriate event handler is called, instrumentation can be implemented in the application's source code to insert alternate instruction sets (i.e. anticipatory Instruction Sets **526**, etc.) at which point the modified source code can be recompiled and/or executed. In some aspects, the state of the application can be saved before recompiling its modified source code so that the application may continue from its prior state. Saving the application's state can be achieved by saving its variables, data structures, objects, location of its current instruction, and/or other necessary information in environmental variables, memory, or other repositories where they can be accessed once the application is recompiled. In other aspects, application's variables, data structures, objects, address of its current instruction, and/or other necessary information can be saved in a repository such as file, database, or other repository accessible to the application after recompilation of its source code. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program **18** can be implemented through modifying or redirecting Application Program's **18** execution path. Generally, an application can be loaded into memory and the flow of execution proceeds from one instruction set to the next until the end of the application. An application may include a branching mechanism that can be driven by keyboard or other input devices, system events, and/or other computing system components or events that may impact the execution path. The execution path can also be altered by an external application through acquiring control of execution and/or redirecting execution to a function, routine/subroutine, or an alternate code segment at any point in the application's execution. A branch, jump, or other mechanism can be utilized to implement the redirected execution. For example, a jump instruction can be inserted at a specific point in an application's execution to redirect execution to an alternate code segment. A jump instruction set may include, for example, an unconditional branch, which always results in branching, or a conditional branch, which may or may not result in branching depending on a condition. When executing an application, a computer may fetch and execute instruction sets in sequence until it encounters a branch instruction set. If the instruction set is an unconditional branch, or it is conditional and the condition is satisfied, the computer may fetch its next instruction set from a different instruction set sequence or code segment as specified by the branch instruction set. After the execution of the alternate code segment, control may be redirected back to the original jump point or to another point in the application. For example, modifying an application can be implemented by redirecting execution of an application to alternate instruction sets (i.e. anticipatory Instruction Sets **526**, etc.). Alternate instruction sets can be pre-compiled, pre-interpreted, or otherwise pre-translated and ready for execution. Alternate instruction sets can also be JIT compiled, JIT interpreted, or otherwise JIT translated before execution. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program **18** can be implemented through assembly language. Assembly language instructions may be directly related with the architecture's machine instructions as previously described. Assembly language can, therefore, be a powerful tool for implementing direct hardware (i.e. processor registers, memory, etc.) access and manipulations as well as access and manipulations of specialized processor features or instructions. Assembly language can also be a powerful tool for implementing low-level embedded systems, real-time systems, interrupt handlers, self or dynamically modifying code, and/or other applications. Specifically, for instance, self or dynamically modifying code that can be used to facilitate modifying of an application can be seamlessly implemented using assembly language. For example, using assembly language, instruc-

tion sets can be dynamically created and loaded into memory similar to the ones that a compiler may generate. Furthermore, using assembly language, memory space of a loaded application can be accessed to modify (including rewrite, overwrite, etc.) original instruction sets or to insert jumps or branches to alternate code elsewhere in memory. Some operating systems may implement protection from changes to applications loaded into memory. Operating system's, processor's, or other low level features or commands can be used to unprotect the protected locations in memory before the change as previously described. Alternatively, a pointer that may reside in a memory location where it could be readily altered can be utilized where the pointer may reference alternate code. In one example, assembly language can be utilized to write alternate code (i.e. anticipatory Instruction Sets **526**, etc.) into a location in memory outside a running application's memory space. Assembly language can then be utilized to redirect the application's execution to the alternate code by inserting a jump or branch into the application's in-memory code, by redirecting program counter, or by other technique. In another example, assembly language can be utilized to overwrite or rewrite the entire or part of an application's in-memory code with alternate code. In some aspects, high-level programming languages can call an external assembly language program to facilitate application modification as previously described. In yet other aspects, relatively low-level programming languages such as C may allow embedding assembly language directly in their source code such as, for example, using asm keyword of C. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program **18** can be implemented through binary rewriting. Binary rewriting tools and/or techniques may modify an application's executable. In some aspects, modification can be minor such as in the case of optimization where the original executable's functionality is kept. In other aspects, modification may change the application's functionality such as by inserting alternate code (i.e. anticipatory Instruction Sets **526**, etc.). Examples of binary rewriting tools include SecondWrite, ATOM, DynamoRIO, Purify, Pin, EEL, DynInst, PLTO, and/or others. Binary rewriting may include disassembly, analysis, and/or modification of target application. Since binary rewriting works directly on machine code executable, it is independent of source language, compiler, virtual machine (if one is utilized), and/or other higher level abstraction layers. Also, binary rewriting tools can perform application modifications without access to original source code. Binary rewriting tools include static rewriters, dynamic rewriters, minimally-invasive rewriters, and/or others. Static binary rewriters can modify an executable when the executable is not in use (i.e. not running). The rewritten executable may then be executed including any new or modified functionality. Dynamic binary rewriters can modify an executable during its execution, thereby enabling modification of an application's functionality at runtime. In some aspects, dynamic rewriters can be used for instrumentation or selective modifications such as insertion of alternate code (i.e. anticipatory Instruction Sets **526**, etc.), and/or for other runtime transformations or modifications. For example, some dynamic rewriters can be configured to intercept an application's execution at indirect control transfers and insert instrumentation or other application modifying code. Minimally-invasive rewriters may keep the original machine code to the greatest extent possible. They support limited modifications such as insertion of jumps into and out of instrumented code. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

Referring to FIG. **30**, in a further example, modifying execution and/or functionality of Processor **11** can be implemented through modification of processor registers, memory, or other computing system components. In some aspects, modifying execution and/or functionality of Processor **11** can be implemented by redirecting Processor's **11** execution to alternate instruction sets (i.e. anticipatory Instruction Sets **526**, etc.). In one example, Program Counter **211** may hold or point to a memory address of the next instruction set that will be executed by Processor **11**. Artificial Intelligence Unit **110** may generate anticipatory Instruction Sets **526** and store them in Memory **12** as previously described. Modification Interface **130** may then change Program Counter **211** to point to the location in Memory **12** where anticipatory Instruction Sets **526** are stored. The anticipatory Instruction Sets **526** can then be fetched from the location in Memory **12** pointed to by the modified Program Counter **211** and loaded into Instruction Register **212** for decoding and execution. Once anticipatory Instruction Sets **526** are executed, Modification Interface **130** may change Program Counter **211** to point to the last instruction set before the redirection or to any other instruction set. In other aspects, anticipatory Instruction Sets **526** can be loaded directly into Instruction Register **212**. As previously described, examples of other processor or computing system components that can be used during an instruction cycle include memory address register (MAR), memory data register (MDR), data registers, address registers, general purpose registers (GPRs), conditional registers, floating point registers (FPRs), constant registers, special purpose registers, machine-specific registers, Register Array **214**, Arithmetic Logic Unit **215**, control unit, and/or other circuits or components. Any of the aforementioned processor registers, memory, or other computing system components can be accessed and/or modified to facilitate the disclosed functionalities. In some embodiments, processor interrupt may be issued to facilitate such access and/or modification. In some designs, modifying execution and/or functionality of Processor **11** can be implemented in a program, combination of programs and hardware, or purely hardware system. Dedicated hardware may be built to perform modifying execution and/or functionality of Processor **11** with marginal or no impact to computing overhead. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

Referring to FIGS. **31A**-**31B**, in a further example, modifying execution and/or functionality of Logic Circuit **250** can be implemented through modification of inputs and/or outputs of Logic Circuit **250**. While Processor **11** includes any type of logic circuit, Logic Circuit **250** is described separately herein to offer additional detail on its functioning. Logic Circuit **250** comprises the functionality for performing logic operations using the circuit's inputs and producing outputs based on the logic operations performed as previously described. In one example, Logic Circuit **250** may perform some logic operations using four input values and produce two output values. Modifying execution and/or functionality of Logic Circuit **250** can be implemented by replacing its input values with anticipatory input values (i.e. anticipatory Instruction Sets **526**, etc.). Artificial Intelligence Unit **110** may generate anticipatory input values as previously described. Modification Interface **130** can then transmit the anticipatory input values to Logic Circuit **250** through the four hardwired connections as shown in FIG.

31A. Modification Interface **130** may use Switches **251** to prevent delivery of any input values that may be sent to Logic Circuit **250** from its usual input source. As such, DCADO Unit **100** may cause Logic Circuit **250** to perform its logic operations using the four anticipatory input values, thereby implementing autonomous Device **98** operation. In another example, Logic Circuit **250** may perform some logic operations using four input values and produce two output values. Modifying execution and/or functionality of Logic Circuit **250** can be implemented by replacing its output values with anticipatory output values (i.e. anticipatory Instruction Sets **526**, etc.). Artificial Intelligence Unit **110** may generate anticipatory output values (i.e. anticipatory Instruction Sets **526**, etc.) as previously described. Modification Interface **130** can then transmit the anticipatory output values through the two hardwired connections as shown in FIG. **31**B. Modification Interface **130** may use Switches **251** to prevent delivery of any output values that may be sent by Logic Circuit **250**. As such, DCADO Unit **100** may bypass Logic Circuit **250** and transmit the two anticipatory output values to downstream elements, thereby implementing autonomous Device **98** operation. In a further example, instead of or in addition to modifying input and/or output values of Logic Circuit **250**, the execution and/or functionality of Logic Circuit **250** may be modified by modifying values or signals in one or more Logic Circuit's **250** internal components such as registers, memories, buses, and/or others (i.e. similar to the previously described modifying of Processor **11** components, etc.). In some designs, modifying execution and/or functionality of Logic Circuit **250** can be implemented in a program, combination of programs and hardware, or purely hardware system. Dedicated hardware may be built to perform modifying execution and/or functionality of Logic Circuit **250** with marginal or no impact to computing overhead. Any of the elements and/or techniques for modifying execution and/or functionality of Logic Circuit **250** can similarly be implemented with Processor **11** and/or other processing elements.

In some embodiments, DCADO Unit **100** may directly modify the functionality of an actuator (previously described, not shown). For example, Logic Circuit **250** or other processing element may control an actuator that enables Device **98** to perform mechanical, physical, and/or other operations. An actuator may receive one or more input values or control signals from Logic Circuit **250** or other processing element directing the actuator to perform specific operations. Modifying functionality of an actuator can be implemented by replacing its input values with anticipatory input values (i.e. anticipatory Instruction Sets **526**, etc.) as previously described with respect to replacing input values of Logic Circuit **250**. Specifically, for instance, Artificial Intelligence Unit **110** may generate anticipatory input values as previously described. Modification Interface **130** can then transmit the anticipatory input values to the actuator. Modification Interface **130** may use Switches **251** to prevent delivery of any input values that may be sent to the actuator from its usual input source. As such, DCADO Unit **100** may cause the actuator to perform its operations using the anticipatory input values, thereby implementing autonomous Device **98** operation.

One of ordinary skill in art will recognize that FIGS. **31**A-**31**B depict one of many implementations of Logic Circuit **250** and that any number of input and/or output values can be utilized in alternate implementations. One of ordinary skill in art will also recognize that Logic Circuit **250** may include any number and/or combination of logic components to implement any logic operations.

Other additional techniques or elements can be utilized as needed for modifying execution and/or functionality of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing elements, or some of the disclosed techniques or elements can be excluded, or a combination thereof can be utilized in alternate embodiments.

Referring to FIG. **32**, the illustration shows an embodiment of a method **9100** for learning and/or using a device's circumstances for autonomous device operation. In some aspects, the method can be used on a computing enabled device or system to enable learning of a device's operation in circumstances including objects with various properties and enable autonomous device operation in similar circumstances. Method **9100** may include any action or operation of any of the disclosed methods such as method **9200**, **9300**, **9400**, **9500**, **9600**, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method **9100**.

At step **9105**, a first collection of object representations is received. A collection of object representations (i.e. Collection of Object Representations **525**, etc.) may include one or more object representations (i.e. Object Representations **625**, etc.), object properties (i.e. Object Properties **630**, etc.), and/or other elements or information. An object representation may include an electronic representation of an object (i.e. Object **615**, etc.) detected in a device's surrounding. In some aspects, a collection of object representations may include one or more object representations, object properties, and/or other elements or information detected in a device's (i.e. Device's **98**, etc.) surrounding at a particular time. A collection of object representations may, therefore, include knowledge (i.e. unit of knowledge, etc.) of a device's circumstances including objects with various properties at a particular time. In some designs, a collection of object representations may include or be associated with a time stamp (not shown), order (not shown), or other time related information. In some embodiments, a collection of object representations may include or be substituted with a stream of collections of object representations, and vice versa. Therefore, the terms collection of object representations and stream of collections of object representations may be used interchangeably herein depending on context. A stream of collections of object representations may include one collection of object representations or a group, sequence, or other plurality of collections of object representations. In some aspects, a stream of collections of object representations may include one or more collections of object representations, and/or other elements or information detected in a device's surrounding over time. A stream of collections of object representations may, therefore, include knowledge (i.e. unit of knowledge, etc.) of a device's circumstances including objects with various properties over time. As circumstances including objects with various properties in a device's surrounding change (i.e. objects and/or their properties change, move, act, transform, etc.) over time, this change may be captured in a stream of collections of object representations. In some designs, each collection of object representations in a stream may include or be associated with the aforementioned time stamp, order, or other time related information. Examples of objects include biological objects (i.e. persons, animals, vegetation, etc.), nature objects (i.e. rocks, bodies of water, etc.), manmade objects (i.e. buildings, streets, ground/aerial/aquatic vehicles, etc.), and/or others. In some aspects, any part of an object may be detected as an object itself. For instance,

instead of or in addition to detecting a vehicle as an object, a wheel and/or other parts of the vehicle may be detected as objects. In general, an object may include any object or part thereof that can be detected. Examples of object properties include existence of an object, type of an object (i.e. person, cat, vehicle, building, street, tree, rock, etc.), identity of an object (i.e. name, identifier, etc.), distance of an object, bearing/angle of an object, location of an object (i.e. distance and bearing/angle from a known point, object coordinates, etc.), shape/size of an object (i.e. height, width, depth, computer model, point cloud, etc.), activity of an object (i.e. motion, gestures, etc.), and/or other properties of an object. In general, an object property may include any attribute of an object (i.e. existence of an object, type of an object, identity of an object, shape/size of an object, etc.), any relationship of an object with the device, other objects, or the environment (i.e. distance of an object, bearing/angle of an object, friend/foe relationship, etc.), and/or other information related to an object. Objects and/or their properties can be detected by one or more sensors (i.e. Sensors **92**, etc.) and/or an object processing unit (i.e. Object Processing Unit **93**, etc.). A sensor may obtain or detect information about its environment. As such, one or more sensors can be used to detect objects and/or their properties in a device's surrounding. In some designs, a sensor may be part of a device whose circumstances are being used for DCADO functionalities. In other designs, a sensor may be part of a remote device whose circumstances are being used for DCADO functionalities. Examples of a sensor include a camera (i.e. Camera **92***a*, etc.), a microphone (i.e. Microphone **92***b*, etc.), a lidar (i.e. Lidar **92***c*, etc.), a radar (i.e. Radar **92***d*, etc.), a sonar (i.e. Sonar **92***e*, etc.), and/or others. An object processing unit may process output from a sensor to obtain information of interest. As such, an object processing unit can be used to process output from a sensor to detect objects and/or their properties in a device's surrounding. In some aspects, an object processing unit may create or generate a collection of object representations. In other aspects, an object processing unit may create or generate a stream of collections of object representations. An object processing unit may include a picture recognizer (i.e. Picture Recognizer **94***a*, etc.), a sound recognizer (i.e. Sound Recognizer **94***b*, etc.), a lidar processing unit (i.e. Lidar Processing Unit **94***c*, etc.), a radar processing unit (i.e. Radar Processing Unit **94***d*, etc.), a sonar processing unit (i.e. Sonar Processing Unit **94***e*, etc.), and/or other elements or functionalities. In general, an object processing unit may include any signal processing element or technique known in art as applicable. In some implementations, an object processing unit and/or any of its elements or functionalities can be included in sensor and/or other elements. Receiving comprises any action or operation by or for a Collection of Object Representations **525**, stream of Collections of Object Representations **525**, Object Representation **625**, Object Property **630**, Sensor **92**, Camera **92***a*, Microphone **92***b*, Lidar **92***c*, Radar **92***d*, Sonar **92***e*, Object Processing Unit **93**, Picture Recognizer **94***a*, Sound Recognizer **94***b*, Lidar Processing Unit **94***c*, Radar Processing Unit **94***d*, Sonar Processing Unit **94***e*, and/or other disclosed elements.

At step **9110**, a first one or more instruction sets for operating a device are received. In some embodiments, an instruction set (i.e. Instruction Set **526**, etc.) may be used or executed by a processor (i.e. Processor **11**, etc.) in operating a device. In other embodiments, an instruction set may be part of an application program (i.e. Application Program **18**, etc.) used in operating a device. For example, the application can run or execute on one or more processors or other processing elements. In further embodiments, an instruction set may be used or executed by a logic circuit (i.e. Logic Circuit **250**, etc.) in operating a device. For example, such instruction set may be or include one or more inputs into or outputs from a logic circuit. In further embodiments, an instruction set may be used by an actuator in operating a device. For example, such instruction set may be or include one or more inputs into an actuator. Operating a device includes performing or causing any operations on/by/with the device. In some designs, an instruction set can be received from a processor, application program, logic circuit, and/or other processing element as the instruction set is being used or executed. In other aspects, an instruction set can be received from a processor, application program, logic circuit, and/or other processing element after the instruction set is used or executed. In further aspects, an instruction set can be received from a processor, application program, logic circuit, and/or other processing element before the instruction set has been used or executed. In further aspects, an instruction set can be received from a running processor, running application program, running logic circuit, and/or other running processing element. As such, an instruction set can be received at runtime. In other designs, an instruction set can be received from an actuator. In further designs, an instruction set can be received from memory (i.e. Memory **12**, etc.), hard drive, or any other storage element or repository. In further designs, an instruction set can be received over a network such as Internet, local area network, wireless network, and/or other network. In further designs, an instruction set can be received by an interface (i.e. Acquisition Interface **120**, etc.) configured to obtain instruction sets from a processor, application program, logic circuit, actuator, and/or other element. In general, an instruction set can be received by any element of the system. One or more instruction sets may temporally correspond to a collection of object representations. In some aspects, an instruction set that temporally corresponds to a collection of object representations includes an instruction set used or executed at the time of generating the collection of object representations. In other aspects, an instruction set that temporally corresponds to a collection of object representations includes an instruction set used or executed within a certain time period before and/or after generating the collection of object representations. Any time period can be utilized depending on implementation. In further aspects, an instruction set that temporally corresponds to a collection of object representations includes an instruction set used or executed from the time of generating the collection of object representations to the time of generating a next collection of object representations. In further aspects, an instruction set that temporally corresponds to a collection of object representations includes an instruction set used or executed from the time of generating a preceding collection of object representations to the time of generating the collection of object representations. Any other temporal relationship or correspondence between collections of object representations and correlated instruction sets can be implemented. In general, one or more instruction sets that temporally correspond to a collection of object representations enable structuring knowledge of a device's operation at or around the time of generating the collection of object representations. Such functionality enables spontaneous or seamless learning of a device's operation in circumstances including objects with various properties as the device is operated in real life situations. In some embodiments, an instruction set may include one or more commands, keywords, symbols (i.e. parentheses, brackets, commas, semicolons, etc.), instructions, operators

(i.e. =, <, >, etc.), variables, values, objects, data structures, functions (i.e. Function1( ), FIRST( ), MIN( ), SQRT( ), etc.), parameters, states, signals, inputs, outputs, references thereto, and/or other components. In other embodiments, an instruction set may include source code, bytecode, intermediate code, compiled, interpreted, or otherwise translated code, runtime code, assembly code, machine code, and/or any other computer code. In further embodiments, an instruction set may include one or more inputs into and/or outputs from a logic circuit. In further embodiments, an instruction set may include one or more inputs into an actuator. Receiving comprises any action or operation by or for an Acquisition Interface **120**, Instruction Set **526**, and/or other disclosed elements.

At step **9115**, the first collection of object representations is correlated with the first one or more instruction sets for operating the device. In some aspects, individual collections of object representations can be correlated with one or more instruction sets. In other aspects, streams of collections of object representations can be correlated with one or more instruction sets. In further aspects, individual collections of object representations or streams of collections of object representations can be correlated with the aforementioned temporally corresponding instruction sets. In further aspects, a collection of object representations or stream of collections of object representations may not be correlated with any instruction sets. Correlating may include structuring or generating a knowledge cell (i.e. Knowledge Cell **800**, etc.) and storing one or more collections of object representations correlated with any instruction sets into the knowledge cell. Therefore, a knowledge cell may include any data structure or arrangement that can facilitate such storing. A knowledge cell includes knowledge (i.e. unit of knowledge, etc.) of how a device operated in a circumstance including objects with various properties. In some designs, extra information (i.e. Extra Info **527**, etc.) may optionally be used to facilitate enhanced comparisons or decision making in autonomous device operation where applicable. Therefore, any collection of object representations, instruction set, and/or other element may include or be correlated with extra information. Extra information may include any information useful in comparisons or decision making performed in autonomous device operation. Examples of extra information include time information, location information, computed information, contextual information, and/or other information. Correlating may be omitted where learning of a device's operation in circumstances including objects with various properties is not implemented. Correlating comprises any action or operation by or for a Knowledge Structuring Unit **520**, Knowledge Cell **800**, and/or other disclosed elements.

At step **9120**, the first collection of object representations correlated with the first one or more instruction sets for operating the device are stored. A collection of object representations correlated with one or more instruction sets may be part of a stored plurality of collections of object representations correlated with one or more instruction sets. Collections of object representations correlated with any instruction sets can be stored in a memory unit or other repository. The aforementioned knowledge cells comprising collections of object representations correlated with any instruction sets can be used in/as neurons, nodes, vertices, or other elements in any of the data structures or arrangements (i.e. neural networks, graphs, sequences, collection of knowledge cells, etc.) used for storing the knowledge of a device's operation in circumstances including objects with various properties. Knowledge cells may be connected, interrelated, or interlinked into knowledge structures using statistical, artificial intelligence, machine learning, and/or other models or techniques. Such interconnected or interrelated knowledge cells can be used for enabling autonomous device operation. The interconnected or interrelated knowledge cells may be stored or organized into a knowledgebase (i.e. Knowledgebase **530**, etc.). In some embodiments, knowledgebase may be or include a neural network (i.e. Neural Network **530***a*, etc.). In other embodiments, knowledgebase may be or include a graph (i.e. Graph **530***b*, etc.). In further embodiments, knowledgebase may be or include a collection of sequences (i.e. Collection of Sequences **530***c*, etc.). In further embodiments, knowledgebase may be or include a sequence (i.e. Sequence **533**, etc.). In further embodiments, knowledgebase may be or include a collection of knowledge cells (i.e. Collection of Knowledge Cells **530***d*, etc.). In general, knowledgebase may be or include any data structure or arrangement, and/or repository capable of storing the knowledge of a device's operation in circumstances including objects with various properties. Knowledgebase may also include or be substituted with various artificial intelligence methods, systems, and/or models for knowledge structuring, storing, and/or representation such as deep learning, supervised learning, unsupervised learning, neural networks (i.e. convolutional neural network, recurrent neural network, deep neural network, etc.), search-based, logic and/or fuzzy logic-based, optimization-based, tree/graph/other data structure-based, hierarchical, symbolic and/or sub-symbolic, evolutionary, genetic, multi-agent, deterministic, probabilistic, statistical, and/or other methods, systems, and/or models. Storing may be omitted where learning of a device's operation in circumstances including objects with various properties is not implemented. Storing comprises any action or operation by or for a Knowledgebase **530**, Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, Knowledge Cell **800**, Node **852**, Layer **854**, Connection **853**, Similarity Comparison **125**, and/or other disclosed elements.

At step **9125**, a new collection of object representations is received. Step **9125** may include any action or operation described in Step **9105** as applicable.

At step **9130**, the new collection of object representations is compared with the first collection of object representations. Comparing one collection of object representations with another collection of object representations may include comparing at least a portion of one collection of object representations with at least a portion of the other collection of object representations. In some embodiments, collections of object representations may be compared individually. In some aspects, comparing of individual collections of object representations may include comparing one or more object representations of one collection of object representations with one or more object representations of another collection of object representations. In other aspects, comparing of object representations may include comparing one or more object properties of one object representation with one or more object properties of another object representation. In some designs, one or more object properties in the same category (i.e. Category **635**, etc.) can be compared. Comparing may include any techniques for comparing text, numbers, and/or other data. In further aspects, some object representations, object properties, and/or other elements of a collection of object representations can be omitted from comparison depending on implementation. In other embodiments, collections of object representations may be compared collectively as part of streams of collections of object representations. Collective comparing of collections of

object representations may include any features, functionalities, and embodiments of the aforementioned individual comparing of collections of object representations. In some aspects, collective comparing of collections of object representations may include comparing one or more collections of object representations of one stream of collections of object representations with one or more collections of object representations of another stream of collections of object representations. In some designs, one or more corresponding (i.e. similarly ordered, temporally related, etc.) collections of object representations from the compared streams of collections of object representations can be compared. In other designs, Dynamic Time Warping (DTW) and/or other techniques can be utilized for comparison and/or aligning temporal sequences (i.e. streams of collections of object representations, etc.) that may vary in time or speed. In further aspects, some collections of object representations can be omitted from comparison depending on implementation. Any combination of the aforementioned and/or other elements or techniques can be utilized in alternate embodiments of the comparing. Comparing may be omitted where anticipating of a device's operation in circumstances including objects with various properties is not implemented. Comparing comprises any action or operation by or for a Decision-making Unit **540**, Similarity Comparison **125**, and/or other disclosed elements.

At step **9135**, a determination is made that there is at least a partial match between the new collection of object representations and the first collection of object representations. In some embodiments, determining at least a partial match between individually compared collections of object representations includes determining that a similarity between one or more portions of one collection of object representations and one or more portions of another collection of object representations exceeds a similarity threshold. In other embodiments, determining at least a partial match between individually compared collections of object representations includes determining at least a partial match between one or more portions of one collection of object representations and one or more portions of another collection of object representations. In further embodiments, determining at least a partial match between individually compared collections of object representations includes determining substantial similarity between one or more portions of one collection of object representations and one or more portions of another collection of object representations. A portion of a collection of object representations may include an object representation, an object property, and/or other portion or element of the collection of object representations. In further embodiments, determining at least a partial match between individually compared collections of object representations includes determining that the number or percentage of matching or substantially matching object representations of the compared collections of object representations exceeds a threshold number (i.e. 1, 2, 4, 7, 18, etc.) or threshold percentage (i.e. 41%, 62%, 79%, 85%, 93%, etc.). In some aspects, type of object representations, importance of object representations, and/or other elements or techniques relating to object representations can be utilized for determining similarity using object representations. In further aspects, some of the object representations can be omitted in determining similarity using object representations depending on implementation. In further embodiments, determining a match or substantial match between compared object representations includes determining that the number or percentage of matching or substantially matching object properties of the compared object representations exceeds a threshold number (i.e. 1, 2, 3, 6, 11, etc.) or a threshold percentage (i.e. 55%, 61%, 78%, 82%, 99%, etc.). In some aspects, categories of object properties, importance of object properties, and/or other elements or techniques relating to object properties can be utilized for determining similarity using object properties. In further aspects, some of the object properties can be omitted in determining similarity using object properties depending on implementation. In some designs, substantial similarity of individually compared collections of object representations can be achieved when a similarity between one or more portions of one collection of object representations and one or more portions of another collection of object representations exceeds a similarity threshold. In other designs, substantial similarity of individually compared collections of object representations can be achieved when the number or percentage of matching or substantially matching object representations of the compared collections of object representations exceeds a threshold number (i.e. 1, 2, 4, 7, 18, etc.) or threshold percentage (i.e. 41%, 62%, 79%, 85%, 93%, etc.). In further aspects, substantial similarity of compared object representations can be achieved when the number or percentage of matching or substantially matching object properties of the compared object representations exceeds a threshold number (i.e. 1, 2, 3, 6, 11, etc.) or a threshold percentage (i.e. 55%, 61%, 78%, 82%, 99%, etc.). In some embodiments, determining at least a partial match between collectively compared collections of object representations (i.e. streams of collections of object representations, etc.) includes determining that the number or percentage of matching or substantially matching collections of object representations of the compared streams of collections of object representations exceeds a threshold number (i.e. 1, 2, 4, 9, 33, 138, etc.) or threshold percentage (i.e. 39%, 58%, 77%, 88%, 94%, etc.). In some aspects, importance of collections of object representations, order of collections of object representations, and/or other elements or techniques relating to collections of object representations can be utilized for determining similarity of collectively compared collections of object representations or streams of collections of object representations. In further aspects, some of the collections of object representations can be omitted in determining similarity of collectively compared collections of object representations or streams of collections of object representations depending on implementation. In some designs, a threshold for a number or percentage similarity can be used to determine a match or substantial match between any of the aforementioned elements. Any text, number, and/or other data similarity determination techniques can be used in any of the aforementioned similarity determinations. A partial match of any of the compared elements may include a substantially or otherwise similar match, and vice versa. Therefore, these terms may be used interchangeably herein depending on context. Although, substantial similarity or substantial match is frequently used herein, it should be understood that any level of similarity, however high or low, may be utilized as defined by the rules (i.e. thresholds, etc.) for similarity. Any combination of the aforementioned and/or other elements or techniques can be utilized in alternate embodiments. Determining may be omitted where anticipating of a device's operation in circumstances including objects with various properties is not implemented. Determining comprises any action or operation by or for a Decision-making Unit **540**, Similarity Comparison **125**, and/or other disclosed elements.

At step **9140**, the first one or more instruction sets for operating the device correlated with the first collection of

object representations are executed. Executing may be performed in response to the aforementioned determining. Executing may be caused by DCADO Unit **100**, Artificial Intelligence Unit **110**, Modification Interface **130**, and/or other disclosed elements. An instruction set may be executed by a processor (i.e. Processor **11**, etc.), application program (i.e. Application Program **18**, etc.), logic circuit (i.e. Logic Circuit **250**, etc.), and/or other processing element. An instruction set may be executed or acted upon by an actuator. In some aspects, instruction sets (i.e. the one or more instruction sets for operating the device correlated with the first collection of object representations, etc.) anticipated or determined to be used or executed in a device's autonomous operation may be referred to as anticipatory instruction sets, alternate instruction sets, and/or other suitable name or reference. Therefore, these terms can be used interchangeably herein depending on context. Executing may include executing one or more alternate instruction sets (i.e. anticipatory instruction sets, etc.) instead of or prior to an instruction set that would have been executed in a regular course of execution. In some embodiments, executing may include modifying a register or other element of a processor with one or more alternate instruction sets. Executing may also include redirecting a processor to one or more alternate instruction sets. In other embodiments, processor may be or comprises a logic circuit. Executing may further include modifying an element of a logic circuit with one or more alternate instruction sets, redirecting the logic circuit to one or more alternate instruction sets, replacing the inputs into the logic circuit with one or more alternate inputs or instruction sets, and/or replacing the outputs from the logic circuit with one or more alternate outputs or instruction sets. Executing may further include replacing the inputs into an actuator with one or more alternate inputs or instruction sets. In further embodiments, a processor may run an application including instruction sets for operating a device. In some aspects, executing includes executing one or more alternate instruction sets as part of the application. In other aspects, executing includes modifying the application. In further aspects, executing includes redirecting the application to one or more alternate instruction sets. In further aspects, executing includes modifying one or more instruction sets of the application. In further aspects, executing includes modifying the application's source code, bytecode, intermediate code, compiled code, interpreted code, translated code, runtime code, assembly code, machine code, or other code. In further aspects, executing includes modifying memory, processor register, storage, repository or other element where the application's instruction sets are stored or used. In further aspects, executing includes modifying instruction sets used for operating an object of the application. In further aspects, executing includes modifying an element of a processor, an element of a device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input used in running the application. In further aspects, executing includes modifying the application at source code write time, compile time, interpretation time, translation time, linking time, loading time, runtime, or other time. In further aspects, executing includes modifying one or more of the application's lines of code, statements, instructions, functions, routines, subroutines, basic blocks, or other code segments. In further aspects, executing includes a manual, automatic, dynamic, just in time (JIT), or other instrumentation of the application. In further aspects, executing includes utilizing one or more of a .NET tool, .NET application programming interface (API), Java tool, Java API, operating system tool, independent tool, or other tool for modifying the application. In further aspects, executing includes utilizing a dynamic, interpreted, scripting, or other programming language. In further aspects, executing includes utilizing dynamic code, dynamic class loading, or reflection. In further aspects, executing includes utilizing assembly language. In further aspects, executing includes utilizing metaprogramming, self-modifying code, or an application modification tool. In further aspects, executing includes utilizing just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further aspects, executing includes utilizing dynamic expression creation, dynamic expression execution, dynamic function creation, or dynamic function execution. In further aspects, executing includes adding or inserting additional code into the application's code. In further aspects, executing includes modifying, removing, rewriting, or overwriting the application's code. In further aspects, executing includes branching, redirecting, extending, or hot swapping the application's code. Branching or redirecting an application's code may include inserting a branch, jump, or other means for redirecting the application's execution. Executing comprises any action or operation by or for a Processor **11**, Application Program **18**, Logic Circuit **250**, Modification Interface **130**, and/or other disclosed elements.

At step **9145**, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations are performed by the device. The one or more operations may be performed in response to the aforementioned executing. In some aspects, an operation includes any operation that can be performed by/with/on a computing enabled device. In other aspects, an operation includes any operation that can be performed by/with/on an actuator. In further aspects, an operation includes any operation that can be performed by/with/on a computer. In general, an operation includes any operation that can be performed by/with/on a device or element thereof. One of ordinary skill in art will recognize that, while all possible variations of operations by/with/on a device are too voluminous to describe and limited only by the device's design and/or user's utilization, all operations are within the scope of this disclosure in various implementations.

Referring to FIG. **33**, the illustration shows an embodiment of a method **9200** for learning and/or using a device's circumstances for autonomous device operation. In some aspects, the method can be used on a computing enabled device or system to enable learning of a device's operation in circumstances including objects with various properties and enable autonomous device operation in similar circumstances. Method **9200** may include any action or operation of any of the disclosed methods such as method **9100**, **9300**, **9400**, **9500**, **9600**, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method **9200**.

At step **9205**, a first collection of object representations is received. Step **9205** may include any action or operation described in Step **9105** of method **9100** as applicable.

At step **9210**, a first one or more instruction sets for operating a device are received. Step **9210** may include any action or operation described in Step **9110** of method **9100** as applicable.

At step **9215**, the first collection of object representations correlated with the first one or more instruction sets for operating the device are learned. Step **9215** may include any

action or operation described in Step **9115** and/or Step **9120** of method **9100** as applicable.

At step **9220**, a new collection of object representations is received. Step **9220** may include any action or operation described in Step **9125** of method **9100** as applicable.

At step **9225**, the first one or more instruction sets for operating the device correlated with the first collection of object representations are anticipated based on at least a partial match between the new collection of object representations and the first collection of object representations. Step **9225** may include any action or operation described in Step **9130** and/or Step **9135** of method **9100** as applicable.

At step **9230**, the first one or more instruction sets for operating the device correlated with the first collection of object representations are executed. Step **9230** may include any action or operation described in Step **9140** of method **9100** as applicable.

At step **9235**, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations are performed by the device. Step **9235** may include any action or operation described in Step **9145** of method **9100** as applicable.

Referring to FIG. **34**, the illustration shows an embodiment of a method **9300** for learning and/or using a device's circumstances for autonomous device operation. In some aspects, the method can be used on a computing enabled device or system to enable learning of a device's operation in circumstances including objects with various properties and enable autonomous device operation in similar circumstances. Method **9300** may include any action or operation of any of the disclosed methods such as method **9100**, **9200**, **9400**, **9500**, **9600**, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method **9300**.

At step **9305**, a first stream of collections of object representations is received. Step **9305** may include any action or operation described in Step **9105** of method **9100** as applicable.

At step **9310**, a first one or more instruction sets for operating a device are received. Step **9310** may include any action or operation described in Step **9110** of method **9100** as applicable.

At step **9315**, the first stream of collections of object representations is correlated with the first one or more instruction sets for operating the device. Step **9315** may include any action or operation described in Step **9115** of method **9100** as applicable.

At step **9320**, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device are stored. Step **9320** may include any action or operation described in Step **9120** of method **9100** as applicable.

At step **9325**, a new stream of collections of object representations is received. Step **9325** may include any action or operation described in Step **9125** of method **9100** as applicable.

At step **9330**, the new stream of collections of object representations is compared with the first stream of collections of object representations. Step **9330** may include any action or operation described in Step **9130** of method **9100** as applicable.

At step **9335**, a determination is made that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. Step **9335** may include any action or operation described in Step **9135** of method **9100** as applicable.

At step **9340**, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations are executed. Step **9340** may include any action or operation described in Step **9140** of method **9100** as applicable.

At step **9345**, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations are performed by the device. Step **9345** may include any action or operation described in Step **9145** of method **9100** as applicable.

Referring to FIG. **35**, the illustration shows an embodiment of a method **9400** for learning and/or using a device's circumstances for autonomous device operation. In some aspects, the method can be used on a computing enabled device or system to enable learning of a device's operation in circumstances including objects with various properties and enable autonomous device operation in similar circumstances. Method **9400** may include any action or operation of any of the disclosed methods such as method **9100**, **9200**, **9300**, **9500**, **9600**, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method **9400**.

At step **9405**, a first collection of object representations is received. Step **9405** may include any action or operation described in Step **9105** of method **9100** as applicable.

At step **9410**, a first one or more inputs are received, wherein the first one or more inputs are also received by a logic circuit, and wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device. Step **9410** may include any action or operation described in Step **9110** of method **9100** as applicable.

At step **9415**, the first collection of object representations is correlated with the first one or more inputs. Step **9415** may include any action or operation described in Step **9115** of method **9100** as applicable.

At step **9420**, the first collection of object representations correlated with the first one or more inputs are stored. Step **9420** may include any action or operation described in Step **9120** of method **9100** as applicable.

At step **9425**, a new collection of object representations is received. Step **9425** may include any action or operation described in Step **9125** of method **9100** as applicable.

At step **9430**, the new collection of object representations is compared with the first collection of object representations. Step **9430** may include any action or operation described in Step **9130** of method **9100** as applicable.

At step **9435**, a determination is made that there is at least a partial match between the new collection of object representations and the first collection of object representations. Step **9435** may include any action or operation described in Step **9135** of method **9100** as applicable.

At step **9440**, the first one or more inputs correlated with the first collection of object representations are received by the logic circuit. Step **9440** may include any action or operation described in Step **9140** of method **9100** as applicable.

At step **9445**, one or more operations defined by one or more outputs for operating the device produced by the logic

circuit are performed by the device. Step **9445** may include any action or operation described in Step **9145** of method **9100** as applicable.

Referring to FIG. **36**, the illustration shows an embodiment of a method **9500** for learning and/or using a device's circumstances for autonomous device operation. In some aspects, the method can be used on a computing enabled device or system to enable learning of a device's operation in circumstances including objects with various properties and enable autonomous device operation in similar circumstances. Method **9500** may include any action or operation of any of the disclosed methods such as method **9100**, **9200**, **9300**, **9400**, **9600**, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method **9500**.

At step **9505**, a first collection of object representations is received. Step **9505** may include any action or operation described in Step **9105** of method **9100** as applicable.

At step **9510**, a first one or more outputs are received, the first one or more outputs transmitted from a logic circuit, wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device. Step **9510** may include any action or operation described in Step **9110** of method **9100** as applicable.

At step **9515**, the first collection of object representations is correlated with the first one or more outputs. Step **9515** may include any action or operation described in Step **9115** of method **9100** as applicable.

At step **9520**, the first collection of object representations correlated with the first one or more outputs are stored. Step **9520** may include any action or operation described in Step **9120** of method **9100** as applicable.

At step **9525**, a new collection of object representations is received. Step **9525** may include any action or operation described in Step **9125** of method **9100** as applicable.

At step **9530**, the new collection of object representations is compared with the first collection of object representations. Step **9530** may include any action or operation described in Step **9130** of method **9100** as applicable.

At step **9535**, a determination is made that there is at least a partial match between the new collection of object representations and the first collection of object representations. Step **9535** may include any action or operation described in Step **9135** of method **9100** as applicable.

At step **9540**, one or more operations defined by the first one or more outputs correlated with the first collection of object representations are performed by the device. Step **9540** may include any action or operation described in Step **9145** of method **9100** as applicable.

Referring to FIG. **37**, the illustration shows an embodiment of a method **9600** for learning and/or using a device's circumstances for autonomous device operation. In some aspects, the method can be used on a computing enabled device or system to enable learning of a device's operation in circumstances including objects with various properties and enable autonomous device operation in similar circumstances. Method **9600** may include any action or operation of any of the disclosed methods such as method **9100**, **9200**, **9300**, **9400**, **9500**, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method **9600**.

At step **9605**, a first collection of object representations is received. Step **9605** may include any action or operation described in Step **9105** of method **9100** as applicable.

At step **9610**, a first one or more inputs are received, wherein the first one or more inputs are also received by an actuator, and wherein the actuator is configured to receive inputs and perform motions. Step **9610** may include any action or operation described in Step **9110** of method **9100** as applicable.

At step **9615**, the first collection of object representations is correlated with the first one or more inputs. Step **9615** may include any action or operation described in Step **9115** of method **9100** as applicable.

At step **9620**, the first collection of object representations correlated with the first one or more inputs are stored. Step **9620** may include any action or operation described in Step **9120** of method **9100** as applicable.

At step **9625**, a new collection of object representations is received. Step **9625** may include any action or operation described in Step **9125** of method **9100** as applicable.

At step **9630**, the new collection of object representations is compared with the first collection of object representations. Step **9630** may include any action or operation described in Step **9130** of method **9100** as applicable.

At step **9635**, a determination is made that there is at least a partial match between the new collection of object representations and the first collection of object representations. Step **9635** may include any action or operation described in Step **9135** of method **9100** as applicable.

At step **9640**, the first one or more inputs correlated with the first collection of object representations are received by the actuator. Step **9640** may include any action or operation described in Step **9140** of method **9100** as applicable.

At step **9645**, one or more motions defined by the first one or more inputs correlated with the first collection of object representations are performed by the actuator. Step **9645** may include any action or operation described in Step **9145** of method **9100** as applicable.

Referring to FIG. **38**, in some exemplary embodiments, Device **98** may be or include Loader **98*a***. Loader **98*a*** may be operated by User **50** in person or remotely. Loader **98*a*** may include or be coupled to one or more Sensors **92** (i.e. collectively referred to as Sensor **92**, etc.) such as Camera **92*a***, Microphone **92*b***, Lidar **92*c***, Radar **92*d***, Sonar **92*e***, etc. and/or Object Processing Unit **93** that can detect Objects **615*aa***-**615*ad***, and/or other elements or information in Loader's **98*a*** surrounding. Object Processing Unit **93** may include Picture Recognizer **94*a***, Sound Recognizer **94*b***, Lidar Processing Unit **94*c***, Radar Processing Unit **94*d***, Sonar Processing Unit **94*e***, and/or other elements or functionalities as applicable. Object Processing Unit **93** may create or generate one or more (i.e. stream, etc.) Collections of Object Representations **525** comprising Object Representations **625**, Object Properties **630**, and/or other elements or information representing Objects **615** detected in Loader's **98*a*** surrounding. Loader **98*a*** may also include or be controlled by Logic Circuit **250** (i.e. microcontroller, etc.), Processor **11** (i.e. including any Application Program **18** running thereon, etc.), and/or other processing element that receives User's **50** (i.e. operator's, etc.) operating directions and causes desired operations with Loader **98*a*** such as moving, maneuvering, collecting, lifting, unloading, and/or others. User **50** can interact with Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element through inputting operating directions via Human-machine Interface **23** such as one or more steering wheels, levers, pedals, buttons, or other input devices. For instance,

responsive to User's **50** manipulating a steering wheel and one or more levers, Logic Circuit **250** or Processor **11** may cause Loader's **98***a* arm with bucket to collect a load, one or more motors or other actuators to move or maneuver Loader **98***a*, lifting system (i.e. hydraulic, pneumatic, mechanical, electrical, etc.) to lift a load, and/or arm with bucket to unload a load. Loader **98***a* may also include or be coupled to DCADO Unit **100**. DCADO Unit **100** may be embedded (i.e. integrated, etc.) into or coupled to Loader's **98***a* Logic Circuit **250**, Processor **11**, and/or other processing element. DCADO Unit **100** may also be a program embedded (i.e. integrated, etc.) into or interfaced with Application Program **18** running on Processor **11** and/or other processing element. DCADO Unit **100** can obtain Instruction Sets **526** used or executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element. In some aspects, Instruction Sets **526** may include one or more inputs into or outputs from Loader's **98***a* Logic Circuit **250** (i.e. microcontroller, etc.). In other aspects, Instruction Sets **526** may include one or more instruction sets from Loader's **98***a* Processor's **11** registers or other components. In further aspects, Instruction Sets **526** may include one or more instruction sets used or executed in Application Program **18**. DCADO Unit **100** may also optionally obtain any Extra Info **527** (i.e. time, location, computed, contextual, and/or other information, etc.) related to Loader's **98***a* operation. As User **50** operates Loader **98***a* in circumstances including objects with various properties as shown, DCADO Unit **100** may learn Loader's **98***a* operations in these circumstances by correlating Collections of Object Representations **525** representing Objects **615** detected in Loader's **98***a* surrounding with one or more Instruction Sets **526** used or executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element. Any Extra Info **527** related to Loader's **98***a* operation may also optionally be correlated with Collections of Object Representations **525**. DCADO Unit **100** may store this knowledge into Knowledgebase **530** (i.e. Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.). In the future, DCADO Unit **110** may compare incoming Collections of Object Representations **525** representing Objects **615** detected in Loader's **98***a* surrounding with previously learned Collections of Object Representations **525** including optionally using any Extra Info **527** for enhanced decision making. If substantially similar or at least a partial match is found or determined, the Instruction Sets **526** correlated with the previously learned Collections of Object Representations **525** can be autonomously executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element, thereby enabling autonomous operation of Loader **98***a* in similar circumstances as in previously learned ones. For instance, Loader **98***a* comprising DCADO Unit **100** may learn User **50**-directed collecting, moving, maneuvering, lifting, unloading, and/or other operations in a circumstance that includes Rock **615***aa*, Pile of Material **615***ab*, Person **615***ac*, Truck **615***ad*, and/or other Objects **615** among which Loader **98***a* may need to maneuver and/or with which Loader **98***a* may need to interact. In the future, when a circumstance that includes Objects **615** with similar Object Properties **630** is encountered, Loader **98***a* may implement collecting, moving, maneuvering, lifting, and/or unloading operations autonomously.

In some embodiments, DCADO Unit **100** may reside on Server **96** accessible over Network **95** as previously described. In such embodiments, any number of Loaders **98***a* may connect to such remote DCADO Unit **100** and the remote DCADO Unit **100** may learn their operations in circumstances including objects with various properties. In turn, any number of Loaders **98***a* can utilize the remote DCADO Unit **100** for autonomous operation in circumstances including objects with various properties. For example, multiple operators (i.e. Users **50**, etc.) may operate their Loaders **98***a* that are configured to transmit their operations in circumstances including objects with various properties to a remote DCADO Unit **100**. Such remote DCADO Unit **100** enables learning of the operators' collective knowledge of operating Loaders **98** in circumstances including objects with various properties. Any number of Loaders **98** can utilize such collective knowledge comprised in the remote DCADO Unit **100** for their autonomous operation. Any of the disclosed elements such as Artificial Intelligence Unit **110**, Knowledgebase **530**, and/or others may reside on Server **96**, and any combination of local and remote elements can be implemented in alternate embodiments.

In some embodiments, Loader **98***a* may include or be coupled to a plurality of Sensors **92** and/or their corresponding Object Processing Units **93**. In one example, multiple Sensors **92** may detect objects and/or their properties from different angles or on different sides of Loader **98***a*. In another example, one or more Sensors **92** may be placed on different sub-devices, sub-systems, or elements of Loader **98***a*. For instance, one Sensor **92** may be placed on the roof of Loader **98***a*, another Sensor **92** may be placed on the arm of Loader **98***a*, and an additional Sensor **92** may be placed on the bucket of Loader **98***a*. In some designs where multiple Sensors **92** are placed on different sub-devices, sub-systems, or elements of Loader **98***a*, multiple DCADO Units **100** can be utilized (i.e. one DCADO Unit **100** for each Sensor **92** or group of Sensors **92** and/or their corresponding Object Processing Units **93**, etc.). In such designs, as User **50** operates Loader **98***a* in circumstances including objects with various properties, a particular DCADO Unit **100** may learn operations of Loader's **98***a* sub-device, sub-system, or element in these circumstances by correlating Collections of Object Representations **525** representing Objects **615** detected by Sensor **92** on the sub-device, sub-system, or element assigned to the DCADO Unit **100** with one or more Instruction Sets **526** used or executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element. The learning and/or decision making in Loader's **98***a* operation can, therefore, be performed per individual sub-device, sub-system, or element. In other designs where multiple Sensors **92** are placed on different sub-devices, sub-systems, or elements of Loader **98***a*, as User **50** operates Loader **98***a* in circumstances including objects with various properties, a single DCADO Unit **100** may learn Loader's **98***a* operations in these circumstances by correlating collective Collections of Object Representations **525** representing Objects **615** detected by Sensors **92** on the sub-devices, sub-systems, or elements with one or more Instruction Sets **526** used or executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element.

In some embodiments, Loader **98***a* may include a plurality of Logic Circuits **250**, Processors **11**, Application Programs **18**, and/or other processing elements. In some aspects, one or more sub-devices, sub-systems, or elements of Loader **98***a* may be controlled by different processing elements. For example, one Processor **11** (i.e. including any Application Programs **18** running thereon, etc.) may control the moving system (i.e. drivetrain, powertrain, etc.) of Loader **98***a*, one Logic Circuit **250** may control an arm of

Loader **98***a*, and an additional Logic Circuit **250** may control a bucket of Loader **98***a*. In some designs where multiple processing elements are utilized, multiple DCADO Units **100** can also be utilized (i.e. one DCADO Unit **100** for each processing element, etc.). In such designs, as User **50** operates Loader **98***a* in circumstances including objects with various properties, a particular DCADO Unit **100** may learn Loader's **98***a* operations in these circumstances by correlating Collections of Object Representations **525** with one or more Instruction Sets **526** used or executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element assigned to the DCADO Unit **100**. The learning and/or decision making in Loader's **98***a* operation can, therefore, be performed per individual processing element. In other designs where multiple processing elements are utilized, as User **50** operates Loader **98***a* in circumstances including objects with various properties, a single DCADO Unit **100** may learn Loader's **98***a* operations in these circumstances by correlating Collections of Object Representations **525** with collective Instruction Sets **526** used or executed by a plurality of Logic Circuits **250**, Processors **11**, Application Programs **18**, and/or other processing elements.

In some embodiments, a combination of DCADO Unit **100** and other systems and/or techniques can be utilized to implement Loader's **98***a* operation. In one example, DCADO Unit **100** may be a primary or preferred system for implementing Loader's **98***a* operation. While operating autonomously under the control of DCADO Unit **100**, Loader **98***a* may encounter a circumstance including objects with various properties that has not been encountered or learned before. In such situations, User **50** and/or non-DCADO system may take control of Loader's **98***a* operation. DCADO Unit **100** may take control again when Loader **98***a* encounters a previously learned circumstance including objects with various properties. Naturally, DCADO Unit **100** can learn Loader's **98***a* operation in the circumstances while User **50** and/or non-DCADO system is in control of Loader **98***a*, thereby reducing or eliminating the need for future involvement of User **50** and/or non-DCADO system. For instance, one User **50** can control or assist in controlling multiple Loaders **98***a* comprising DCADO Units **100**. In such instances, User **50** can control or assist in controlling a Loader **98***a* that may encounter a circumstance including objects with various properties that has not been encountered or learned before while the Loaders **98***a* operating in previously learned circumstances can operate autonomously. In another example, User **50** and/or non-DCADO system may be a primary or preferred system for implementing Loader's **98***a* operation. While operating under the control of User **50** and/or non-DCADO system, User **50** and/or non-DCADO system may release control to DCADO Unit **100** for any reason (i.e. User **50** gets tired or distracted, non-DCADO system gets stuck or cannot make a decision, etc.), at which point Loader **98***a* can be controlled by DCADO Unit **100**. In some designs, DCADO Unit **100** may take control in certain special circumstances including objects with various properties where DCADO Unit **100** may offer superior performance even though User **50** and/or non-DCADO system may generally be preferred. Once Loader **98***a* leaves such special circumstances, DCADO Unit **100** may release control to User **50** and/or non-DCADO system. In general, DCADO Unit **100** can take control from, share control with, or release control to User **50**, non-DCADO system, and/or other system or process at any time, in any circumstances, and remain in control for any period of time as needed.

In some embodiments, DCADO Unit **100** may control one or more sub-devices, sub-systems, or elements of Loader **98***a* while User **50** and/or non-DCADO system may control other one or more sub-devices, sub-systems, or elements of Loader **98***a*. For example, User **50** and/or non-DCADO system may control the moving system (i.e. drivetrain, powertrain, etc.) of Loader **98***a*, while DCADO Unit **100** may control an arm and bucket of Loader **98***a*. Any other combination of controlling various sub-devices, sub-systems, or elements of Loader **98***a* by DCADO Unit **100** and User **50** and/or non-DCADO system can be implemented.

Referring to FIG. **39**, in some exemplary embodiments, Device **98** may be or include Boat **98***b*. Boat **98***b* may be operated by User **50** in person or remotely. Boat **98***b* may include or be coupled to one or more Sensors **92** (i.e. collectively referred to as Sensor **92**, etc.) such as Camera **92***a*, Microphone **92***b*, Lidar **92***c*, Radar **92***d*, Sonar **92***e*, etc. and/or Object Processing Unit **93** that can detect Objects **615***ba*-**615***bd*, and/or other elements or information in Boat's **98***b* surrounding. Object Processing Unit **93** may include Picture Recognizer **94***a*, Sound Recognizer **94***b*, Lidar Processing Unit **94***c*, Radar Processing Unit **94***d*, Sonar Processing Unit **94***e*, and/or other elements or functionalities as applicable. Object Processing Unit **93** may create or generate one or more (i.e. stream, etc.) Collections of Object Representations **525** comprising Object Representations **625**, Object Properties **630**, and/or other elements or information representing Objects **615** detected in Boat's **98***b* surrounding. Boat **98***b* may also include or be controlled by Logic Circuit **250** (i.e. microcontroller, etc.), Processor **11** (i.e. including any Application Program **18** running thereon, etc.), and/or other processing element that receives User's **50** (i.e. operator's, etc.) operating directions and causes desired operations with Boat **98***b* such as moving, maneuvering, and/or other operations. User **50** can interact with Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element through inputting operating directions via Human-machine Interface **23** such as one or more steering wheels, levers, pedals, buttons, or other input devices. For instance, responsive to User's **50** manipulating a steering wheel and one or more levers, Logic Circuit **250** or Processor **11** may cause one or more motors or other actuators to move or maneuver Boat **98***b*. Boat **98***b* may also include or be coupled to DCADO Unit **100**. DCADO Unit **100** may be embedded (i.e. integrated, etc.) into or coupled to Boat's **98***b* Logic Circuit **250**, Processor **11**, and/or other processing element. DCADO Unit **100** may also be a program embedded (i.e. integrated, etc.) into or interfaced with Application Program **18** running on Processor **11** and/or other processing element. DCADO Unit **100** can obtain Instruction Sets **526** used or executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element. In some aspects, Instruction Sets **526** may include one or more inputs into or outputs from Boat's **98***b* Logic Circuit **250** (i.e. microcontroller, etc.). In other aspects, Instruction Sets **526** may include one or more instruction sets from Boat's **98***b* Processor's **11** registers or other components. In further aspects, Instruction Sets **526** may include one or more instruction sets used or executed in Application Program **18**. DCADO Unit **100** may also optionally obtain any Extra Info **527** (i.e. time, location, computed, contextual, and/or other information, etc.) related to Boat's **98***b* operation. As User **50** operates Boat **98***b* in circumstances including objects with various properties as shown, DCADO Unit **100** may learn Boat's **98***b* operations in these circumstances by correlating Collections of Object

Representations **525** representing Objects **615** detected in Boat's **98***b* surrounding with one or more Instruction Sets **526** used or executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element. Any Extra Info **527** related to Boat's **98***b* operation may also optionally be correlated with Collections of Object Representations **525**. DCADO Unit **100** may store this knowledge into Knowledgebase **530** (i.e. Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.). In the future, DCADO Unit **110** may compare incoming Collections of Object Representations **525** representing Objects **615** detected in Boat's **98***b* surrounding with previously learned Collections of Object Representations **525** including optionally using any Extra Info **527** for enhanced decision making. If substantially similar or at least a partial match is found or determined, the Instruction Sets **526** correlated with the previously learned Collections of Object Representations **525** can be autonomously executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element, thereby enabling autonomous operation of Boat **98***b* in similar circumstances as in previously learned ones. For instance, Boat **98***b* comprising DCADO Unit **100** may learn User **50**-directed moving, maneuvering, and/or other operations in a circumstance that includes Fishing Boat **615***ba*, Lighthouse **615***bb*, Sailboat **615***bc*, Cruise Ship **615***bd*, and/or other Objects **615** among which Boat **98***b* may need to maneuver. In the future, when a circumstance that includes Objects **615** with similar Object Properties **630** is encountered, Boat **98***b* may implement moving, maneuvering, and/or other operations autonomously. In some aspects, the shore (not enumerated) or any part thereof (i.e. cliff, ridge, beach, etc.) may be detected as an Object **615** itself, which may then be learned and used in autonomous operation of Boat **98***b*.

Referring to FIG. **40**, in some exemplary embodiments, an Area of Interest **450** can be utilized. In one example, Area of Interest **450** may include a radial, circular, elliptical, or other such area around Boat **98***b*. In another example, Area of Interest **450** may include a triangular, rectangular, octagonal, or other such area around Boat **98***b*. In a further example, Area of Interest **450** may include a spherical, cubical, pyramid-like, or other such area around Boat **98***b* as applicable to 3D space. Any other Area of Interest **450** shape can be utilized depending on implementation. The shape and/or size of Area of Interest **450** can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. Utilizing Area of Interest **450** enables DCADO Unit **100** to focus on Boat's **98***b* immediate surrounding, thereby avoiding extraneous detail in the rest of the surrounding. In some aspects, Area of Interest **450** can be subdivided into sub-areas (i.e. sub-circles, sub-rectangles, sub-spheres, etc.). Sub-areas can be used to classify the surrounding by distance from Boat **98***b*. For example, the surrounding closer to Boat **98***b* may be more important and may be assigned higher importance index or weight. As User **50** operates Boat **98***b* in circumstances including objects with various properties as shown, DCADO Unit **100** may learn Boat's **98***b* operations in these circumstances by correlating Collections of Object Representations **525** representing Objects **615** detected in Area of Interest **450** around Boat **98***b* with one or more Instruction Sets **526** used or executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element. Any Extra Info **527** related to Boat's **98***b* operation may also optionally be correlated with Collections of Object Representations **525**. DCADO Unit **100** may store this knowledge into Knowledgebase **530** (i.e. Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.). In the future, DCADO Unit **110** may compare incoming Collections of Object Representations **525** representing Objects **615** detected in Area of Interest **450** around Boat **98***b* with previously learned Collections of Object Representations **525** including optionally using any Extra Info **527** for enhanced decision making. If substantially similar or at least a partial match is found or determined, the Instruction Sets **526** correlated with the previously learned Collections of Object Representations **525** can be autonomously executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element, thereby enabling autonomous operation of Boat **98***b* in similar Areas of Interest **450** as in previously learned ones. For instance, Boat **98***b* comprising DCADO Unit **100** may learn User **50**-directed moving, maneuvering, and/or other operations in an Area of Interest **450** that includes Fishing Boat **615***ba*, Lighthouse **615***bb*, Cruise Ship **615***bd*, and/or other Objects **615** among which Boat **98***b* may need to maneuver. In the future, when an Area of Interest **450** that includes Objects **615** with similar Object Properties **630** is encountered, Boat **98***b* may implement moving, maneuvering, and/or other operations autonomously.

The features, functionalities, and embodiments described with respect to Loader **98***a* and Boat **98***b* can be implemented in any situation where Device **98** may need to autonomously maneuver among, interact with, or perform other operations relative to objects in its surrounding. Therefore, the features, functionalities, and embodiments described with respect to Loader **98***a* and Boat **98***b* can similarly be implemented on any computing enabled machine such as a bulldozer, an excavator, a crane, a forklift, a truck, a construction machine, an assembly machine, an object handling machine, an object dispensing machine, a sorting machine, a restocking machine, an industrial machine, an agricultural machine, a harvesting machine, a building control system, a home or other appliance, a toy, a robot, a tank, an aircraft, a vessel, a submarine, a ground vehicle, an aerial vehicle, an aquatic vehicle, and/or other computing-enabled machine or system.

In yet some exemplary embodiments, Device **98** may be or include a control device such as a thermostat, control panel, remote or other controller, and/or other control device. For instance, a thermostat comprising DCADO Unit **100** may learn User's **50** setting temperature of an air conditioning system controlled by the thermostat in a circumstance that includes User **50** and/or other persons entering or being present in a room. In the future, when a circumstance that includes User **50** and/or other persons entering or being present in the room is encountered, thermostat may implement setting temperature of the air conditioning system autonomously. In some aspects, a control device may be included in the device being controlled (i.e. control panel of an oven, refrigerator, fixture, etc.). In other aspects, a control device may be separate from the device being controlled (i.e. remote controller of a television device, etc.). In yet further exemplary embodiments, Device **98** may be or include a mobile computer such as a smartphone, tablet, and/or other mobile computer. For instance, a smartphone comprising DCADO Unit **100** may learn User **50**-directed playing a music file, setting a vibrate mode, and/or other operations in a circumstance that includes objects with various properties. In the future, when a circumstance that includes objects with similar properties is

encountered, smartphone may implement playing music file, setting vibrate mode, and/or other operations autonomously. In general, Device **98** may be or include any movable, stationary, or other device. One of ordinary skill in art will understand that Device **98** may be or include any device that can implement and/or benefit from the functionalities described herein.

It must be noted that as used herein and in the appended claims, the singular forms "a", "an", and "the" include plural referents unless the context clearly dictates otherwise.

A number of embodiments have been described herein. While this disclosure contains many specific implementation details, these should not be construed as limitations on the scope of the disclosure or of what may be claimed, but rather as descriptions of features specific to particular embodiments. It should be understood that various modifications can be made without departing from the spirit and scope of the disclosure. The logic flows depicted in the figures do not require the particular order shown, or sequential order, to achieve desirable results. In addition, other or additional steps, elements, or connections can be included, or some of the steps, elements, or connections can be eliminated, or a combination thereof can be utilized in the described flows, illustrations, or descriptions. Further, the various aspects of the disclosed devices, apparatuses, systems, and/or methods can be combined in whole or in part with each other to produce additional implementations. Moreover, separation of various components in the embodiments described herein should not be understood as requiring such separation in all embodiments, and it should be understood that the described components can generally be integrated together in a single product or packaged into multiple products. Accordingly, other embodiments are within the scope of the following claims.

The invention claimed is:

1. A method implemented using a computing system that includes one or more processor circuits, the method comprising:

accessing a memory that stores at least a knowledgebase that includes: a first correlation including a first circumstance representation correlated with a first one or more instruction sets for operating a first device and a second correlation including a second circumstance representation correlated with a second one or more instruction sets for operating the first device, wherein the first circumstance representation represents a first circumstance detected at least in part by one or more sensors of the first device and the second circumstance representation represents a second circumstance detected at least in part by the one or more sensors of the first device, and wherein at least a portion of the first correlation and at least a portion of the second correlation are learned in a learning process that includes operating the first device at least partially by a user;

generating or receiving a third circumstance representation, wherein the third circumstance representation represents a third circumstance detected at least in part by the one or more sensors of the first device or at least in part by one or more sensors of a second device;

anticipating the first one or more instruction sets for operating the first device learned in the learning process based on at least partial match between the third circumstance representation and the first circumstance representation; and

at least in response to the anticipating, executing the first one or more instruction sets for operating the first device learned in the learning process, wherein the first device or the second device autonomously performs one or more operations defined by the first one or more instruction sets for operating the first device learned in the learning process.

2. The method of claim **1**, wherein the knowledgebase further includes: a first knowledge cell and a second knowledge cell, and wherein the first knowledge cell includes the first correlation and the second knowledge cell includes the second correlation.

3. The method of claim **1**, wherein the learning process includes: creating, inserting, deleting, modifying, or manipulating an element of the first correlation, or creating, inserting, deleting, modifying, or manipulating an element of the second correlation.

4. The method of claim **1**, wherein the learning process includes:

generating or receiving the first circumstance representation, and generating or receiving the second circumstance representation; and

obtaining or receiving the first one or more instruction sets for operating the first device, and obtaining or receiving the second one or more instruction sets for operating the first device, and wherein the knowledgebase further includes a fourth correlation including a fourth circumstance representation correlated with a fourth one or more instruction sets for operating the second device, and wherein the fourth circumstance representation represents a fourth circumstance detected at least in part by the one or more sensors of the second device, and wherein at least a portion of the fourth correlation is learned in another learning process that includes:

generating or receiving the fourth circumstance representation; and

obtaining or receiving the fourth one or more instruction sets for operating the second device.

5. The method of claim **1**, wherein the anticipating the first one or more instruction sets for operating the first device learned in the learning process based on the at least partial match between the third circumstance representation and the first circumstance representation includes:

determining that a number of at least partially matching portions of the third circumstance representation and portions of the first circumstance representation exceeds a threshold number, or

determining that a percentage of at least partially matching portions of the third circumstance representation and portions of the first circumstance representation exceeds a threshold percentage.

6. The method of claim **1**, wherein the at least the portion of the first correlation and the at least the portion of the second correlation are learned in the learning process while the user operates the first device, and wherein the one or more operations defined by the first one or more instruction sets for operating the first device learned in the learning process correspond to the user's methodology of operating the first device learned in the learning process.

7. The method of claim **1**, wherein the knowledgebase further includes a fourth correlation including a fourth circumstance representation correlated with a fourth one or more instruction sets for operating: the first device, the second device, or a third device, and wherein a first connection is generated to connect the first correlation with the second correlation, and wherein a second connection is generated to connect the second correlation with the fourth correlation.

8. The method of claim **1**, further comprising:

modifying the first one or more instruction sets for operating the first device learned in the learning process or a copy of the first one or more instruction sets for operating the first device learned in the learning process, and wherein the executing the first one or more instruction sets for operating the first device learned in the learning process includes executing the modified the first one or more instruction sets for operating the first device learned in the learning process or the modified the copy of the first one or more instruction sets for operating the first device learned in the learning process, and wherein the performing, by the first device or by the second device, the one or more operations defined by the first one or more instruction sets for operating the first device learned in the learning process includes performing, by the first device or by the second device, one or more operations defined by the modified the first one or more instruction sets for operating the first device learned in the learning process or by the modified the copy of the first one or more instruction sets for operating the first device learned in the learning process.

9. The method of claim **1**, wherein the knowledgebase further includes a fourth correlation including a fourth circumstance representation correlated with a fourth one or more instruction sets for operating the first device, and wherein the fourth circumstance representation represents a fourth circumstance detected at least in part by the one or more sensors of the first device, and wherein at least a portion of the fourth correlation is learned in another learning process that includes operating the first device at least partially by the user.

10. The method of claim **1**, wherein the knowledgebase further includes a fourth correlation including a fourth circumstance representation correlated with a fourth one or more instruction sets for operating the first device, and wherein the fourth circumstance representation represents a fourth circumstance detected at least in part by the one or more sensors of the first device, and wherein at least a portion of the fourth correlation is learned in another learning process that includes operating the first device at least partially by another user.

11. The method of claim **1**, wherein the knowledgebase further includes a fourth correlation including a fourth circumstance representation correlated with a fourth one or more instruction sets for operating the second device, and wherein the fourth circumstance representation represents a fourth circumstance detected at least in part by the one or more sensors of the second device, and wherein at least a portion of the fourth correlation is learned in another learning process that includes operating the second device at least partially by another user.

12. The method of claim **1**, wherein the first circumstance includes one or more objects detected at least in part by the one or more sensors of the first device at a first time or during a first time period, and wherein the second circumstance includes one or more objects detected at least in part by the one or more sensors of the first device at a second time or during a second time period, and wherein the third circumstance includes:

one or more objects detected at least in part by the one or more sensors of the first device at a third time or during a third time period, or

one or more objects detected at least in part by the one or more sensors of the second device at a third time or during a third time period.

13. The method of claim **1**, wherein the first circumstance representation is a data structure including one or more data about the first circumstance of the first device, and wherein the second circumstance representation is a data structure including one or more data about the second circumstance of the first device, and wherein the third circumstance representation is a data structure including one or more data about: the third circumstance of the first device, or the third circumstance of the second device.

14. The method of claim **1**, wherein the first circumstance representation includes: one or more object representations, or one or more collections of object representations, and wherein the second circumstance representation includes: one or more object representations, or one or more collections of object representations, and wherein the third circumstance representation includes: one or more object representations, or one or more collections of object representations.

15. The method of claim **1**, wherein, to correlate the first circumstance representation with the first one or more instruction sets for operating the first device, a determination is made that the first one or more instruction sets for operating the first device temporally correspond to the first circumstance representation, and wherein the first circumstance representation includes: one or more object representations, or one or more collections of object representations.

16. The method of claim **1**, wherein the anticipating the first one or more instruction sets for operating the first device learned in the learning process based on the at least partial match between the third circumstance representation and the first circumstance representation includes determining the first one or more instruction sets for operating the first device learned in the learning process based on the at least partial match between the third circumstance representation and the first circumstance representation.

17. The method of claim **1**, wherein elements of the computing system are included in: a single device, or multiple devices, and wherein the one or more processor circuits include: one or more microcontrollers, one or more computing circuits, or one or more electronic circuits, and wherein the memory includes: a volatile memory, or a non-volatile memory, and wherein the first device includes: a robot, a vehicle, an appliance, an electronic device, or a mechanical machine, and wherein the second device includes: a robot, a vehicle, an appliance, an electronic device, or a mechanical machine, and wherein an instruction set of the first one or more instruction sets for operating the first device includes at least one selected from the group comprising: only one instruction, a plurality of instructions, one or more inputs, one or more commands, one or more computer commands, one or more keywords, one or more symbols, one or more operators, one or more variables, one or more values, one or more objects or object references, one or more data structures or data structure references, one or more functions or function references, one or more parameters, one or more signals, one or more characters, one or more digits, one or more user operating directions, one or more user directions, one or more user inputs, one or more representations of one or more user actions, one or more representations of one or more user clicks, one or more binary bits, one or more assembly language commands, one or more states, one or more codes, one or more data, and one or more information, and wherein an instruction set of the second one or more instruction sets for operating the first device includes at least one selected from the group comprising: only one instruction, a plurality of instructions, one or more inputs, one or more commands, one or more

computer commands, one or more keywords, one or more symbols, one or more operators, one or more variables, one or more values, one or more objects or object references, one or more data structures or data structure references, one or more functions or function references, one or more parameters, one or more signals, one or more characters, one or more digits, one or more user operating directions, one or more user directions, one or more user inputs, one or more representations of one or more user actions, one or more representations of one or more user clicks, one or more binary bits, one or more assembly language commands, one or more states, one or more codes, one or more data, and one or more information, and wherein the one or more sensors of the first device include at least one selected from the group comprising: one or more cameras, one or more microphones, one or more radars, one or more lidars, one or more sonars, and one or more apparatuses for detecting objects or object properties, and wherein the one or more sensors of the second device include at least one selected from the group comprising: one or more cameras, one or more microphones, one or more radars, one or more lidars, one or more sonars, and one or more apparatuses for detecting objects or object properties, and wherein the at least the portion of the first correlation includes: one portion of the first correlation, multiple portions of the first correlation, all portions of the first correlation, or the entire first correlation, and wherein the at least the portion of the second correlation includes: one portion of the second correlation, multiple portions of the second correlation, all portions of the second correlation, or the entire second correlation, and wherein an object of the first circumstance is the same as an object of the third circumstance, or multiple objects of the first circumstance are the same as multiple objects of the third circumstance, or all objects of the first circumstance are the same as all objects of the third circumstance, or all objects of the first circumstance are different than all objects of the third circumstance.

18. One or more non-transitory machine readable media storing machine readable code that when executed by one or more processor circuits causes the one or more processor circuits to perform at least:

accessing a memory that stores at least a knowledgebase that includes: a first correlation including a first circumstance representation correlated with a first one or more instruction sets for operating a first device and a second correlation including a second circumstance representation correlated with a second one or more instruction sets for operating the first device, wherein the first circumstance representation represents a first circumstance detected at least in part by one or more sensors of the first device and the second circumstance representation represents a second circumstance detected at least in part by the one or more sensors of the first device, and wherein at least a portion of the first correlation and at least a portion of the second correlation are learned in a learning process that includes operating the first device at least partially by a user;

generating or receiving a third circumstance representation, wherein the third circumstance representation represents a third circumstance detected at least in part by the one or more sensors of the first device or at least in part by one or more sensors of a second device;

anticipating the first one or more instruction sets for operating the first device learned in the learning process based on at least partial match between the third circumstance representation and the first circumstance representation; and

at least in response to the anticipating, causing the first device or the second device to perform one or more operations defined by the first one or more instruction sets for operating the first device learned in the learning process at least by causing the one or more processor circuits or another one or more processor circuits to execute the first one or more instruction sets for operating the first device learned in the learning process.

19. A system comprising:

a memory that stores at least a knowledgebase that includes: a first correlation including a first circumstance representation correlated with a first one or more instruction sets for operating a first device and a second correlation including a second circumstance representation correlated with a second one or more instruction sets for operating the first device, wherein the first circumstance representation represents a first circumstance detected at least in part by one or more sensors of the first device and the second circumstance representation represents a second circumstance detected at least in part by the one or more sensors of the first device, and wherein at least a portion of the first correlation and at least a portion of the second correlation are learned in a learning process that includes operating the first device at least partially by a user;

means for generating or receiving a third circumstance representation, wherein the third circumstance representation represents a third circumstance detected at least in part by the one or more sensors of the first device or at least in part by one or more sensors of a second device;

means for anticipating the first one or more instruction sets for operating the first device learned in the learning process based on at least partial match between the third circumstance representation and the first circumstance representation; and

means for executing, at least in response to the anticipating, the first one or more instruction sets for operating the first device learned in the learning process, wherein the first device or the second device autonomously.

20. The system of claim **19**, wherein the means for generating or receiving the third circumstance representation includes one or more processor circuits, and wherein the means for anticipating the first one or more instruction sets for operating the first device learned in the learning process based on the at least partial match between the third circumstance representation and the first circumstance representation includes one or more processor circuits, and wherein the means for executing, at least in response to the anticipating, the first one or more instruction sets for operating the first device learned in the learning process includes one or more processor circuits.

* * * * *

Exhibit B

US011238344B1

(12) **United States Patent**
Cosic

(10) **Patent No.: US 11,238,344 B1**
(45) **Date of Patent: *Feb. 1, 2022**

(54) **ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION**

(71) Applicant: **Jasmin Cosic**, Miami, FL (US)

(72) Inventor: **Jasmin Cosic**, Miami, FL (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 182 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: **16/540,972**

(22) Filed: **Aug. 14, 2019**

Related U.S. Application Data

(63) Continuation of application No. 15/340,991, filed on Nov. 2, 2016, now Pat. No. 10,452,974.

(51) **Int. Cl.**
G06N 3/08 (2006.01)
G06N 5/02 (2006.01)

(52) **U.S. Cl.**
CPC ***G06N 3/08*** (2013.01); ***G06N 5/022*** (2013.01)

(58) **Field of Classification Search**
None
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,370,707 A 1/1983 Phillips et al.
4,730,315 A 3/1988 Saito et al.
4,860,203 A 8/1989 Corrigan et al.
5,560,011 A 9/1996 Uyama
5,602,982 A 2/1997 Judd et al.
6,026,234 A 2/2000 Hanson et al.
6,088,731 A 7/2000 Kiraly et al.
6,106,299 A 8/2000 Ackermann et al.
6,126,330 A 10/2000 Knight
6,314,558 B1 11/2001 Angel et al.
6,408,262 B1 * 6/2002 Leerberg G06F 11/3608 703/2
6,643,842 B2 11/2003 Angel et al.
6,728,689 B1 4/2004 Drissi et al.
6,735,632 B1 5/2004 Kiraly et al.
6,754,631 B1 6/2004 Din
6,801,912 B2 10/2004 Moskowitz et al.
6,842,877 B2 1/2005 Robarts et al.
6,915,105 B2 7/2005 Masuda
6,973,446 B2 12/2005 Mamitsuka et al.
7,017,153 B2 3/2006 Gouriou et al.
7,052,277 B2 5/2006 Kellman
7,113,946 B2 9/2006 Cosic
7,117,225 B2 10/2006 Cosic
7,222,127 B1 5/2007 Bem et al.
(Continued)

OTHER PUBLICATIONS

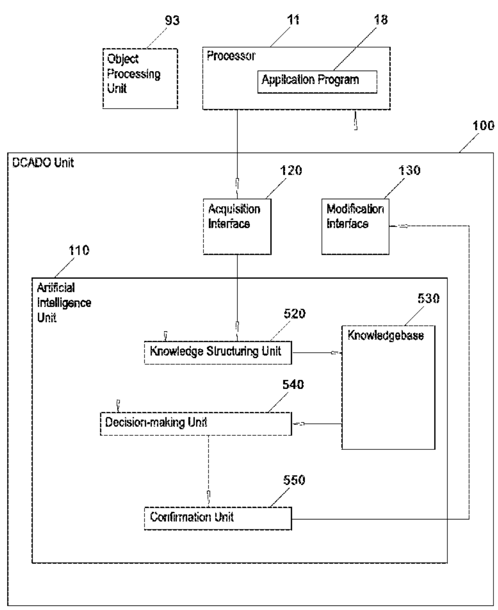
ABOUT | OpenCV, retrieved from <URL: http://opencv.org/about.html> on Dec. 13, 2014, 1 pages.
(Continued)

Primary Examiner — Santiago Garcia

(57) **ABSTRACT**

Aspects of the disclosure generally relate to computing enabled devices and/or systems, and may be generally directed to devices, systems, methods, and/or applications for learning a device's operation in various circumstances, storing this knowledge in a knowledgebase (i.e. neural network, graph, sequences, etc.), and enabling autonomous operation of the device.

20 Claims, 40 Drawing Sheets



(56) **References Cited**

U.S. PATENT DOCUMENTS

7,240,335 B2 7/2007 Angel et al.
7,249,349 B2 7/2007 Hundt et al.
7,424,705 B2 9/2008 Lewis et al.
7,478,371 B1 1/2009 Gove
7,484,205 B2 1/2009 Venkatapathy
7,565,340 B2 7/2009 Herlocker et al.
7,721,218 B2 5/2010 Awe et al.
7,765,537 B2 7/2010 Havin et al.
7,797,259 B2 9/2010 Jiang et al.
7,840,060 B2 11/2010 Podilchuk
7,925,984 B2 4/2011 Awe et al.
7,987,144 B1 7/2011 Drissi et al.
8,005,828 B2 8/2011 Buchner et al.
8,019,699 B2 9/2011 Baxter
8,078,556 B2 12/2011 Adi et al.
8,090,669 B2 1/2012 Shahani et al.
8,137,112 B2 3/2012 Woolf et al.
8,166,463 B2 4/2012 Gill et al.
8,195,674 B1 6/2012 Bem et al.
8,196,119 B2 6/2012 Gill et al.
8,244,730 B2 8/2012 Gupta
8,251,704 B2 8/2012 Woolf et al.
8,261,199 B2 9/2012 Cradick et al.
8,266,608 B2 9/2012 Hecht et al.
8,335,805 B2 12/2012 Cosic
8,356,026 B2 1/2013 Heimendinger
8,364,612 B2 1/2013 Van Gael et al.
8,386,401 B2 2/2013 Virkar et al.
8,397,227 B2 3/2013 Fan et al.
8,417,740 B2 4/2013 Cosic
8,464,225 B2 6/2013 Greifeneder
8,549,359 B2 10/2013 Zheng
8,572,035 B2 10/2013 Cosic
8,589,414 B2 11/2013 Waite et al.
8,595,154 B2 * 11/2013 Breckenridge G06N 20/00 706/12
8,655,260 B2 2/2014 Yueh et al.
8,655,900 B2 2/2014 Cosic
8,667,472 B1 3/2014 Molinari
8,996,432 B1 3/2015 Fu
9,047,324 B2 6/2015 Cosic
9,158,813 B2 10/2015 Ntoulas et al.
9,268,454 B2 2/2016 Hamilton, II et al.
9,282,309 B1 * 3/2016 Cosic H04N 9/87
9,298,749 B2 3/2016 Cosic
9,305,216 B1 4/2016 Mishra
9,367,806 B1 * 6/2016 Cosic G06N 5/022
9,443,192 B1 * 9/2016 Cosic G06N 3/02
9,488,984 B1 11/2016 Williams et al.
9,582,762 B1 * 2/2017 Cosic G06N 5/04
9,864,933 B1 * 1/2018 Cosic G06K 9/66
10,102,226 B1 * 10/2018 Cosic G06F 16/5866
10,102,449 B1 * 10/2018 Cosic G06K 9/00771
10,162,740 B1 12/2018 Setty et al.
2003/0026588 A1 2/2003 Elder et al.
2003/0065662 A1 4/2003 Cosic
2004/0117771 A1 6/2004 Venkatapathy
2004/0194017 A1 9/2004 Cosic
2004/0249774 A1 12/2004 Caid et al.
2004/0267521 A1 12/2004 Cutler et al.
2005/0149517 A1 7/2005 Cosic
2005/0149542 A1 7/2005 Cosic
2005/0240412 A1 10/2005 Fujita
2005/0245303 A1 11/2005 Graepel et al.
2005/0289105 A1 12/2005 Cosic
2006/0047612 A1 3/2006 Stanley et al.
2006/0184410 A1 8/2006 Ramamurthy et al.
2006/0190930 A1 8/2006 Hecht et al.
2006/0265406 A1 11/2006 Chkodrov et al.
2007/0006159 A1 1/2007 Hecht et al.
2007/0050606 A1 3/2007 Ferren et al.
2007/0050719 A1 3/2007 Lui et al.
2007/0058856 A1 3/2007 Boregowda et al.
2007/0061735 A1 3/2007 Hoffberg et al.
2007/0106633 A1 5/2007 Reiner
2008/0144893 A1 6/2008 Guo et al.
2008/0215508 A1 9/2008 Hanneman et al.
2008/0254429 A1 10/2008 Woolf et al.
2008/0281764 A1 11/2008 Baxter
2008/0288259 A1 11/2008 Chambers et al.
2009/0044113 A1 2/2009 Jones et al.
2009/0067727 A1 3/2009 Hirohata
2009/0110061 A1 4/2009 Lee et al.
2009/0131152 A1 5/2009 Busse
2009/0136095 A1 5/2009 Marcon et al.
2009/0141969 A1 6/2009 Yu et al.
2009/0222388 A1 9/2009 Hua et al.
2009/0287643 A1 11/2009 Corville et al.
2009/0324010 A1 12/2009 Hou
2010/0023541 A1 1/2010 Cosic
2010/0033780 A1 2/2010 Gitter
2010/0063949 A1 3/2010 Eaton et al.
2010/0082536 A1 4/2010 Cosic
2010/0114746 A1 5/2010 Bobbitt et al.
2010/0138370 A1 6/2010 Wu et al.
2010/0241595 A1 9/2010 Felsher
2010/0278420 A1 * 11/2010 Shet G06K 9/00369 382/156
2011/0007079 A1 1/2011 Perez et al.
2011/0030031 A1 2/2011 Lussier et al.
2011/0085734 A1 4/2011 Berg et al.
2011/0218672 A1 9/2011 Maisonnier et al.
2011/0270794 A1 11/2011 Drory et al.
2012/0150773 A1 6/2012 DiCorpo et al.
2012/0167057 A1 * 6/2012 Schmich G06F 11/3644 717/130
2012/0192096 A1 7/2012 Bowman et al.
2012/0284026 A1 11/2012 Cardillo et al.
2012/0290347 A1 * 11/2012 Elazouni G06Q 10/00 705/7.12
2013/0007532 A1 1/2013 Miller et al.
2013/0156345 A1 6/2013 Shmunk
2013/0159021 A1 6/2013 Felsher
2013/0218932 A1 8/2013 Cosic
2013/0226974 A1 8/2013 Cosic
2013/0238533 A1 9/2013 Virkar et al.
2013/0278501 A1 10/2013 Bulzacki
2013/0278631 A1 10/2013 Border et al.
2013/0329946 A1 12/2013 Archibald et al.
2014/0052717 A1 2/2014 Franks et al.
2014/0075249 A1 3/2014 Sato et al.
2014/0161250 A1 6/2014 Leeds et al.
2014/0177946 A1 * 6/2014 Lim G06K 9/4614 382/156
2014/0207580 A1 6/2014 Minnis et al.
2014/0211988 A1 7/2014 Fan et al.
2014/0278754 A1 9/2014 Cronin et al.
2015/0006171 A1 1/2015 Westby et al.
2015/0023186 A1 * 1/2015 Vasseur H04L 43/103 370/252
2015/0023205 A1 * 1/2015 Vasseur H04L 45/42 370/254
2015/0039304 A1 2/2015 Wein
2015/0055821 A1 2/2015 Fotland
2015/0089081 A1 * 3/2015 Thubert H04W 40/30 709/239
2015/0264306 A1 9/2015 Marilly et al.
2015/0269156 A1 9/2015 Awadallah et al.
2015/0269415 A1 9/2015 Gelbman
2015/0310041 A1 10/2015 Kier et al.
2015/0324685 A1 * 11/2015 Bohn G06N 3/04 706/15
2015/0339213 A1 * 11/2015 Lee G06F 11/3664 717/125
2016/0063118 A1 3/2016 Campbell et al.
2016/0077166 A1 * 3/2016 Morozov G01C 19/00 702/150
2016/0140999 A1 5/2016 Cosic
2016/0142650 A1 5/2016 Cosic
2016/0167226 A1 * 6/2016 Schnittman B25J 5/00 382/153
2016/0243701 A1 * 8/2016 Gildert G05B 19/42
2016/0246819 A1 8/2016 Cosic

(56) **References Cited**

U.S. PATENT DOCUMENTS

2016/0246850	A1	8/2016	Cosic	
2016/0246868	A1	8/2016	Cosic	
2016/0274187	A1*	9/2016	Menon	G01R 31/31718
2016/0292185	A1	10/2016	Cosic	
2016/0328480	A1*	11/2016	Owens	G06Q 30/02
2017/0161958	A1*	6/2017	Eilat	G02B 27/017
2018/0017791	A1*	1/2018	Beckman	G02B 27/0103
2018/0060144	A1*	3/2018	Capobianco	G06F 9/542
2018/0068564	A1	3/2018	Tanigawa et al.	
2018/0096261	A1*	4/2018	Chu	G06N 7/005
2018/0101146	A1*	4/2018	Hariharan	G05B 13/0265
2018/0120813	A1*	5/2018	Coffman	G05B 19/4097
2018/0174207	A1*	6/2018	Potucek	G06Q 30/0283
2018/0375720	A1*	12/2018	Yang	G06N 20/00
2019/0028545	A1*	1/2019	Yang	H04W 4/00
2019/0034765	A1	1/2019	Kaehler et al.	
2019/0107839	A1	4/2019	Parashar et al.	
2019/0213446	A1*	7/2019	Tsou	H04W 4/70

OTHER PUBLICATIONS

Animetrics, Inc. | 3D Facial Recognition, retrieved from <URL: http://animetrics.com/> on Dec. 13, 2014, 2 pages.
Bag-of-words model, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 2 pages.
Beier-Neely morphing algorithm, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 1 pages.
Computer vision, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 7 pages.
Convolutional neural network, retrieved from <URL: http://wikipedia.com> on Nov. 11, 2015, 5 pages.
Decimation (signal processing), retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 3 pages.
Digital image processing, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 3 pages.
Dissolve (filmmaking), retrieved from <URL: http://wikipedia.com> on Nov. 11, 2015, 2 pages.
Facial recognition system, retrieved from <URL: http://wikipedia.com> on Nov. 11, 2015, 6 pages.
Feature detection (computer vision), retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 3 pages.
Feature extraction, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 2 pages.
Gesture recognition, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 5 pages.
GrabCut, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 1 pages.
Image processing, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 2 pages.
Image segmentation, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 12 pages.
Image warping, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 2 pages.
Inbetweening, retrieved from <URL: http://wikipedia.com> on Nov. 11, 2015, 2 pages.
Interpolation, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 5 pages.
Language model, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 4 pages.
List of speech recognition software, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 4 pages.
Livewire Segmentation Technique, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 2 pages.
Morphing, retrieved from <URL: http://wikipedia.com> on Nov. 11, 2015, 2 pages.
Motion estimation, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 2 pages.
Muse: Face Morph || Mesh Warping, retrieved from <URL: http://alexwolfe.blogspot.com/2011/10/face-morph-mesh-warping.html> on Nov. 19, 2015, 2 pages.
N-gram, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 5 pages.
Optical flow, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 4 pages.
Outline of object recognition, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 7 pages.
Phonotactics, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 3 pages.
Recurrent neural network, retrieved from <URL: http://wikipedia.com> on Nov. 11, 2015, 5 pages.
Sample rate conversion, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 2 pages.
Simple interactive object extraction, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 2 pages.
Speech recognition, retrieved from <URL: http://wikipedia.com> on Nov. 11, 2015, 12 pages.
Speech segmentation, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 4 pages.
Welcome to recognize-speech.com, retrieved from <URL: http://recognize-speech.com/> on Oct. 18, 2015, 1 pages.
Introduction Speech, retrieved from <URL: http://recognize-speech.com/speech> on Oct. 18, 2015, 1 pages.
Preprocessing, retrieved from <URL: http://recognize-speech.com/preprocessing> on Oct. 18, 2015, 4 pages.
Feature Extraction, retrieved from <URL: http://recognize-speech.com/feature-extraction> on Oct. 18, 2015, 3 pages.
Acoustic model, retrieved from <URL: http://recognize-speech.com/acoustic-model> on Oct. 18, 2015, 2 pages.
Video content analysis, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 2 pages.
Video tracking, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 2 pages.
Andrej Karpathy, Thomas Leung, George Toderici, Rahul Sukthankar, Sanketh Shetty, Li Fei-Fei, Large-scale Video Classification with Convolutional Neural Networks, Apr. 14, 2014, 8 pages, Stanford University.
Karen Simonyan, Andrew Zisserman, Two-Stream Convolutional Networks for Action Recognition in Videos, Nov. 13, 2014, 11 pages, University of Oxford.
Tracing (software), retrieved from <URL: http://wikipedia.com> on Jan. 10, 2014, 3 pages.
Tree (data structure), retrieved from <URL: http://wikipedia.com> on Jun. 24, 2014, 6 pages.
PTRACE(2), retrieved from <URL: http://unixhelp.ed.ac.uk/CGI/man-cgi?ptrace> on Mar. 19, 2014, 5 pages.
Wevtutil, retrieved from <URL: http://technet.microsoft.com/en-us/library/cc732848(d=default,l=en-us,v=ws.11).aspx> on Apr. 28, 2014, 5 pages.
Intel Processor Trace, retrieved from <URL: https://software.intel.com/en-us/blogs/2013/09/18/processor-tracing> on Apr. 28, 2014, 3 pages.
Younghoon Jung, Java Dynamics Reflection and a lot more, Oct. 10, 2012, 55 pages, Columbia University.
Amitabh Srivastava, Alan Eustace, Atom A System for Building Customized Program Analysis Tools, May 3, 2004, 12 pages.
Mathew Smithson, Kapil Anand, Aparna Kotha, Khaled Elwazeer, Nathan Giles, Rajeev Barua, Binary Rewriting without Relocation Information, Nov. 10, 2010, 11 pages, University of Maryland.
Marek Olszewski, Keir Mierte, Adam Czajkowski, Angela Demle Brown, JIT Instrumentation—A Novel Approach to Dynamically Instrument Operating Systems, Feb. 12, 2007, 14 pages, University of Toronto.
About Event Tracing, retrieved from <URL: http://msdn.microsoft.com/en-us/library/aa363668(d=default,l=en-us,v=vs.85).aspx> on Jan. 12, 2014, 2 pages.
Add dynamic Java code to your application, retrieved from <URL: http://www.javaworld.com/article/2071777/design-patterns/add-dynamic-java-code-to-your-application.html> on May 13, 2014, 6 pages.
BCEL, retrieved from <URL: http://commons.apache.org> on May 13, 2014, 2 pages.
Artificial intelligence, retrieved from <URL: http://wikipedia.com> on May 21, 2014, 28 pages.

(56) **References Cited**

OTHER PUBLICATIONS

Artificial neural network, retrieved from <URL: http://wikipedia.com> on May 21, 2014, 12 pages.
Branch (computer science), retrieved from <URL: http://wikipedia.com> on May 6, 2014, 2 pages.
Branch trace, retrieved from <URL: http://wikipedia.com> on May 28, 2014, 2 pages.
C Function Call Conventions and the Stack, retrieved from <URL: http://www.csee.umbc.edu/~chang/cs313.s02/stack.shtml> on Jan. 11, 2014, 4 pages.
Call stack, retrieved from <URL: http://wikipedia.com> on Jan. 11, 2014, 5 pages.
Program compilation and execution flow, retrieved from <URL: http://cs.stackexchange.com/questions/6187/program-compilation-and-execution-flow> on Jan. 9, 2014, 2 pages.
Control unit, retrieved from <URL: http://wikipedia.com> on May 25, 2014, 2 pages.
Create dynamic applications with javax.tools, retrieved from <URL: http://www.ibm.com/developerworks/library/j-jcomp/> on May 13, 2014, 11 pages.
Creating a Debugging and Profiling Agent with JVMTI, retrieved from <URL: http://www.oracle.com/technetwork/articles/java/jvmti-136367.html> on Jan. 11, 2014, 18 pages.
Decision tree learning, retrieved from <URL: http://wikipedia.com> on Jun. 24, 2014, 5 pages.
Dynamic Bytecode Instrumentation, retrieved from <URL: http://www.drdobbs.com/tools/dynamic-bytecode-instrumentation/184406433> on Apr. 26, 2014, 6 pages.
Dynamic programming language, retrieved from <URL: http://wikipedia.com> on May 7, 2014, 4 pages.
Dynamic recompilation, retrieved from <URL: http://wikipedia.com> on May 19, 2014, 4 pages.
Eval(), retrieved from <URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/eval> on May 7, 2014, 13 pages.
Function, retrieved from <URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Function> on May 7, 2014, 11 pages.
FunctionEnter2 Function, retrieved from <URL: https://msdn.microsoft.com/en-us/library/aa964981(d=default,l=en-us,v=vs.110).aspx> on Mar. 19, 2014, 2 pages.
FunctionLeave2 Function, retrieved from <URL: https://msdn.microsoft.com/en-us/library/aa964942(d=default,l=en-us,v=vs.110).aspx> on Mar. 19, 2014, 1 pages.
Heuristic, retrieved from <URL: http://wikipedia.com> on May 22, 2014, 5 pages.
How to: Configure Trace Switches, retrieved from <URL: https://msdn.microsoft.com/en-us/library/vstudio/t06xyy08(d=default,l=en-us,v=vs.100).aspx> on Mar. 19, 2014, 3 pages.
Indirect branch, retrieved from <URL: http://wikipedia.com> on May 5, 2014, 1 pages.
Instruction cycle, retrieved from <URL: http://wikipedia.com> on Apr. 30, 2014, 2 pages.
Instruction set, retrieved from <URL: http://wikipedia.com> on Apr. 29, 2014, 6 pages.
Interrupt, retrieved from <URL: http://wikipedia.com> on May 15, 2014, 5 pages.
Introduction to Instrumentation and Tracing, retrieved from <URL: http://msdn.microsoft.com/en-us/library/aa983649(d=default,l=en-us,v=vs.71).aspx> on Jan. 10, 2014, 2 pages.
Java bytecode, retrieved from <URL: http://wikipedia.com> on May 27, 2014, 4 pages.
Package java.util.logging, retrieved from <URL: http://docs.oracle.com/javase/1.5.0/docs/api/java/util/logging/package-summary.html> on Mar. 19, 2014, 2 pages.
Iavac—Java programming language compiler, retrieved from <URL: http://docs.oracle.com/javase/7/docs/technotes/tools/windows/javac.html> on May 13, 2014, 12 pages.
Interface JavaCompiler, retrieved from <URL: http://wikipedia.com> on May 13, 2014, 4 pages.
Javassist, retrieved from <URL: http://wikipedia.com> on May 13, 2014, 2 pages.
Just-in-time compilation, retrieved from <URL: http://wikipedia.com> on May 19, 2014, 5 pages.
JVM Tool Interface, retrieved from <URL: http://docs.oracle.com/javase/7/docs/platform/jvmti/jvmti.html> on Apr. 26, 2014, 127 pages.
Machine code, retrieved from <URL: http://wikipedia.com> on Apr. 30, 2014, 3 pages.
Metaprogramming, retrieved from <URL: http://wikipedia.com> on May 8, 2014, 3 pages.
Microsoft.VisualBasic.Logging Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/ms128122(d=default,l=en-us,v=vs.110).aspx> on Jan. 13, 2014, 1 pages.
Pin (computer program), retrieved from <URL: http://wikipedia.com> on May 6, 2014, 3 pages.
Processor register, retrieved from <URL: http://wikipedia.com> on Apr. 29, 2014, 4 pages.
Profiling (computer programming), retrieved from <URL: http://wikipedia.com> on Jan. 10, 2014, 4 pages.
Profiling Overview, retrieved from <URL: http://msdn.microsoft.com/en-us/library/bb384493(d=default,l=en-us,v=vs.110).aspx> on Jan. 12, 2014, 6 pages.
Program counter, retrieved from <URL: http://wikipedia.com> on Apr. 24, 2014, 2 pages.
Ptrace, retrieved from <URL: http://wikipedia.com> on Apr. 24, 2014, 2 pages.
Reflection (computer programming), retrieved from <URL: http://wikipedia.com> on May 7, 2014, 5 pages.
Class Runtime, retrieved from <URL: http://docs.oracle.com/javase/7/docs/api/java/lang/Runtime.html> on Jan. 10, 2014, 10 pages.
Self-modifying code, retrieved from <URL: http://wikipedia.com> on Jan. 12, 2014, 6 pages.
Subjective logic, retrieved from <URL: http://wikipedia.com> on May 22, 2014, 5 pages.
System.CodeDom.Compiler Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/z6b99ydt(d=default,l=en-us,v=vs.110).aspx> on May 13, 2014, 2 pages.
System.Reflection.Emit Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/xd5fw18y(d=default, l=en-us,v=vs.110).aspx> on May 13, 2014, 2 pages.
Chen et al. Case-Based Reasoning System and Artificial Neural Networks: A Review Neural Comput & Applic (2001) 10: pp. 264-276, 13 pages.
John J. Grefenstette, Connie Loggia Ramsey, Alan C. Schultz, Learning Sequential Decision Rules Using Simulation Models and Competition, 1990, Navy Center for Applied Research in Artificial Intelligence, Naval Research Laboratory, Washington, DC, 27 pages.
Alan C. Schultz, John J. Grefenstette, Using a Genetic Algorithm to Learn Behaviors for Autonomous Vehicles, 1992, Navy Center for Applied Research in Artificial Intelligence, Naval Research Laboratory, Washington, DC, 12 pages.
Koppula et al., "Anticipating human activities using object affordances for reactive robotic response", IEEE TRAMI 2016, published May 5, 2015, 16 pages.
Orme, "System design tips for entry level smartphones—part 3", found online at "https://community.arm.eom/processors/b/blog/posts/system-design-tips-for-entry-level-smartphones—part-3", Oct. 21, 2013, 12 pages.
Nowostawski, Mariusz et al., "Dynamic Demes Parallel Genetic Algorithm", May 13, 1999, 6 pages.
Stack Overflow, How can I quantify difference between two images?, accessed Aug. 2, 2017 at https://stackoverflow.com/questions/189943/how-can-i-quantify-difference-between-two-images, 8 pages.
Skolicki, Zbigniew et al., "The Influence of Migration Sizes and Intervals on Island Models", Jun. 29, 2005, 8 pages.
Vahid Lari, et al., "Decentralized dynamic resource management support for massively parallel processor arrays", Sep. 11, 2011, 8 pages.
Vahid Lari, et al., "Distributed resource reservation in massively parallel processor arrays", May 16, 2011, 4 pages.

(56) **References Cited**

OTHER PUBLICATIONS

Mohsen Hayati and Yazdan Shirvany, "Artificial Neural Network Approach for Short Term Load Forecasting for Illam Region", Jan. 2007, 5 pages.
Hasim Sak, Andrew Senior, and Francoise Beaufays, "Long Short-Term Memory Recurrent Neural Network Architectures for Large Scale Acoustic Modeling", Jan. 2014, 5 pages.
Jorg Walter and Klaus Schulten, "Implementation of self-organizing neural networks for visuo-motor control of an industrial robot", Jan. 1993, 10 pages.
Heikki Hyotyniemi and Aarno Lehtola, "A Universal Relation Database Interface for Knowledge Based Systems", Apr. 1991, 5 pages.
Mrissa, Michael, et al. "An avatar architecture for the web of things." IEEE Internet Computing 19.2 (2015): 30-38., 9 pages.
Luck, Michael, and Ruth Aylett. "Applying artificial intelligence to virtual reality: Intelligent virtual environments." Applied Artificial Intelligence 14.1 (2000): 3-32., 30 pages.
Terdjimi, Mehdi, et al. "An avatar-based adaptation workflow for the web of things." Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), 2016 IEEE 25th International Conference on. IFFF, 2016., 6 pages.
Bogdanovych, Anton, et al. "Authentic interactive reenactment of cultural heritage with 3D virtual worlds and artificial intelligence." Applied Artificial Intelligence 24.6 (2010): 617-647., 32 pages.
Hernandez, Marco E. Perez, and Stephan Reiff-Marganiec. "Autonomous and self controlling smart objects for the future internet." Future internet of things and cloud (FiCloud), 2015 3rd international conference on. IEEE, 2015., 8 pages.
Medini et al., "Building a Web of Things with Avatars", Managing the Web of Things (2017), 30 pages.
Zhang, Yiying, Lei Guo, and Nicolas D. Georganas. "AGILE: An architecture for agent-based collaborative and interactive virtual environments." Proc. Workshop on Application Virtual Reality Technologies for Future Telecommunication System, IEEE Globecom '2000 Conference. 2000. (Year: 2000), 9 pages.
Eladhari, Mirjam P., "Semi-Autonomous Avatars in Virtual Game Worlds," Pre-conference to the ECREA 2010—3rd European Communication Conference, Avatars and Humans. Representing Users in Digital Games, Hamburg, Germany, Oct. 2010, Available from: Jan. 21, 2012 (Year: 2012), 8 pages.

* cited by examiner

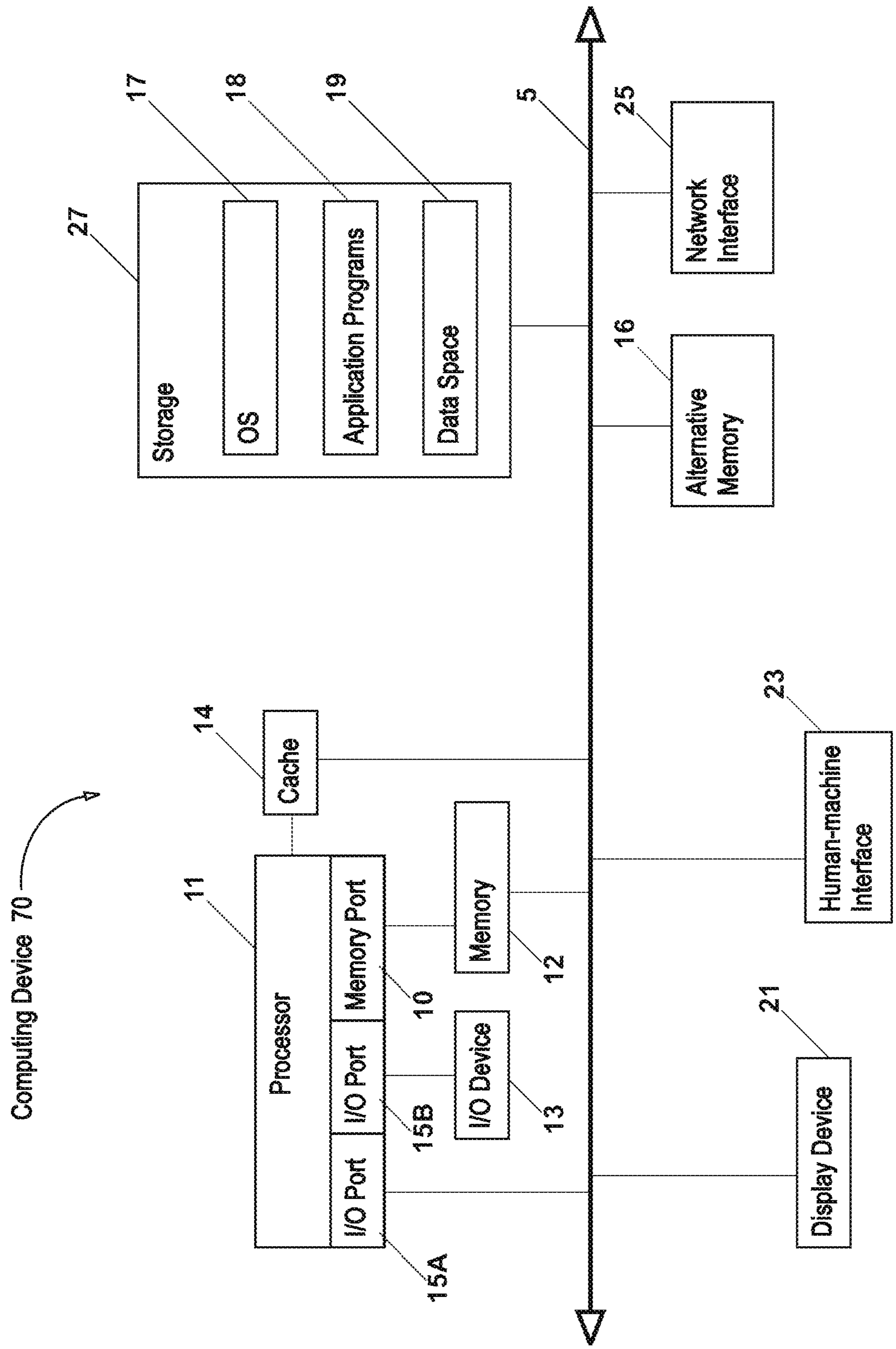

Computing Device 70
Storage
OS
Application Programs
Data Space
27
17
18
19
5
25
Network Interface
16
Alternative Memory
14
Cache
11
Processor
Memory Port
I/O Port
I/O Port
10
15B
15A
Memory
12
I/O Device
13
23
Human-machine Interface
21
Display Device

FIG. 1

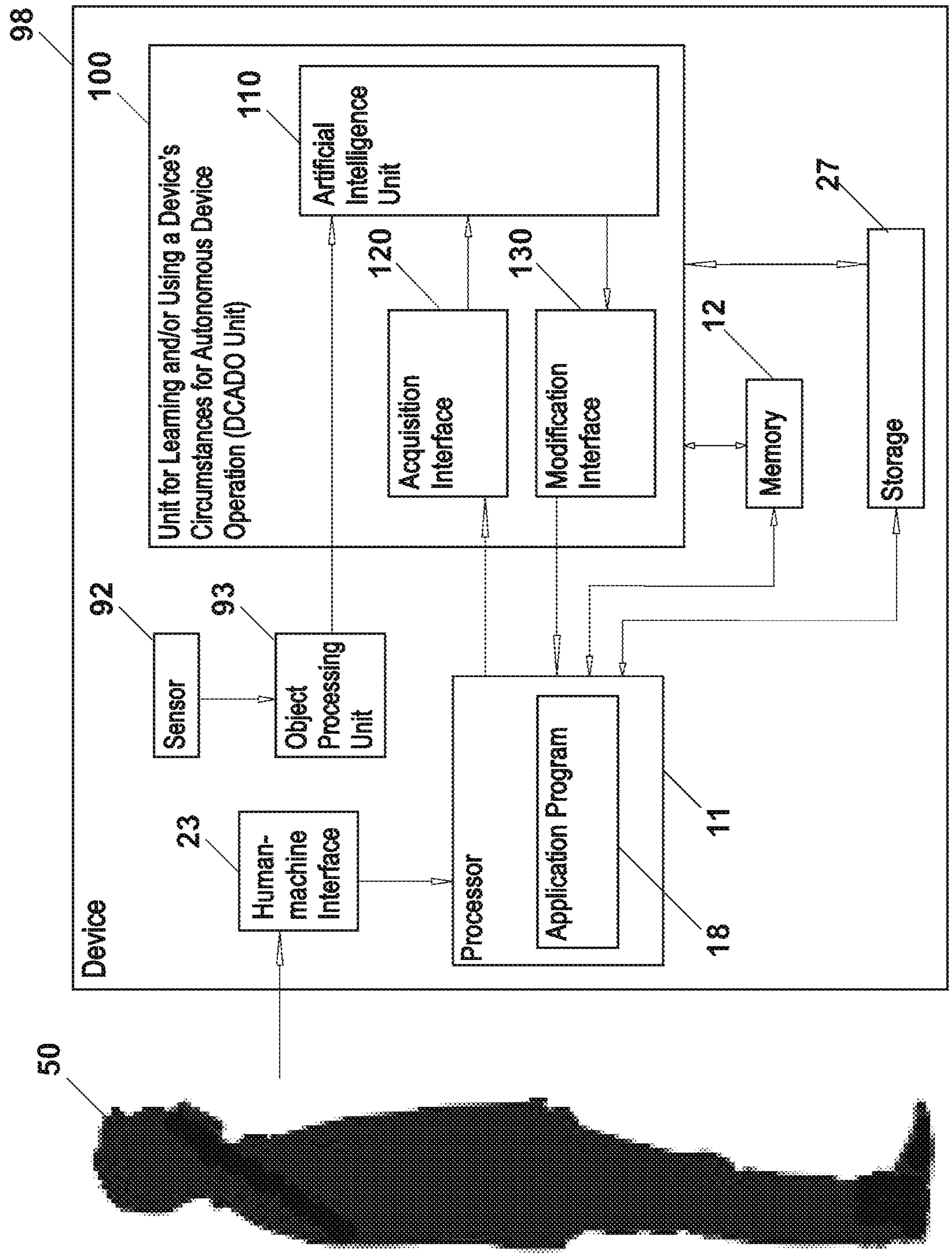

98
Device
100
Unit for Learning and/or Using a Device's Circumstances for Autonomous Device Operation (DCADO Unit)
110
Artificial Intelligence Unit
120
Acquisition Interface
130
Modification Interface
12
Memory
27
Storage
92
Sensor
93
Object Processing Unit
23
Human-machine Interface
Processor
Application Program
11
18
50

FIG. 2

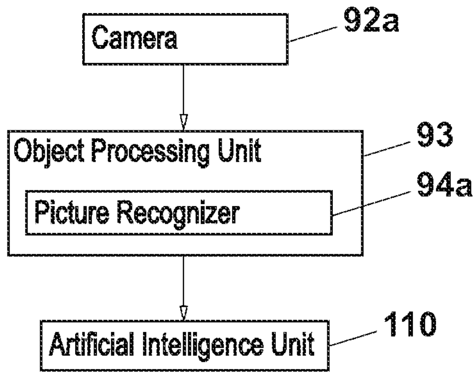


FIG. 3A

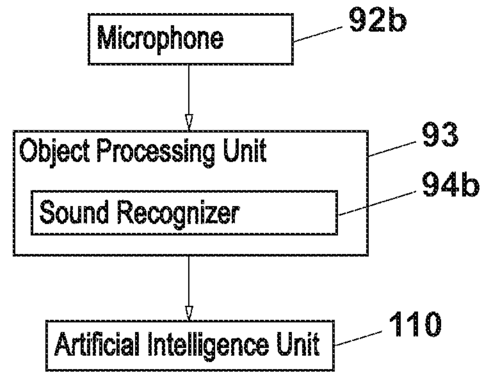


FIG. 3B

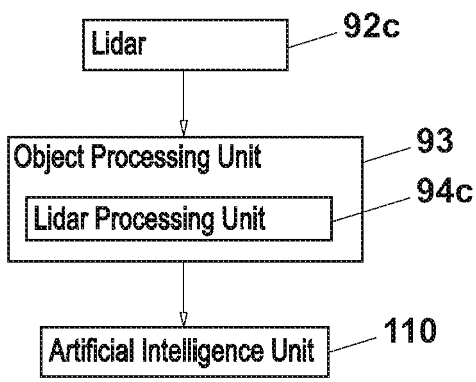


FIG. 3C

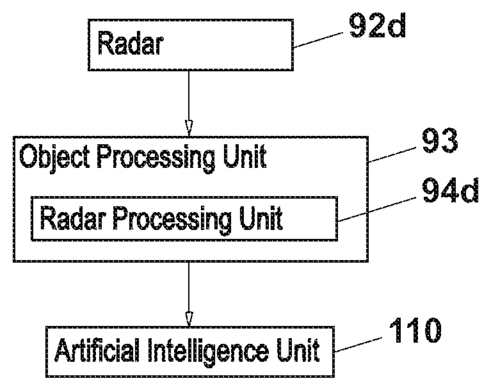


FIG. 3D

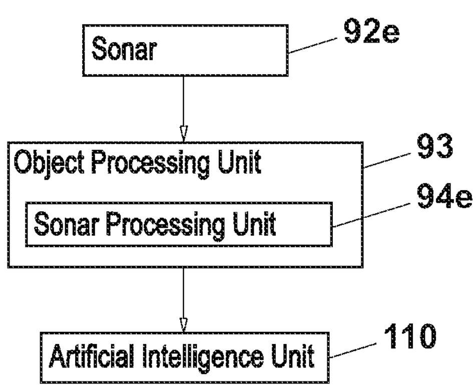
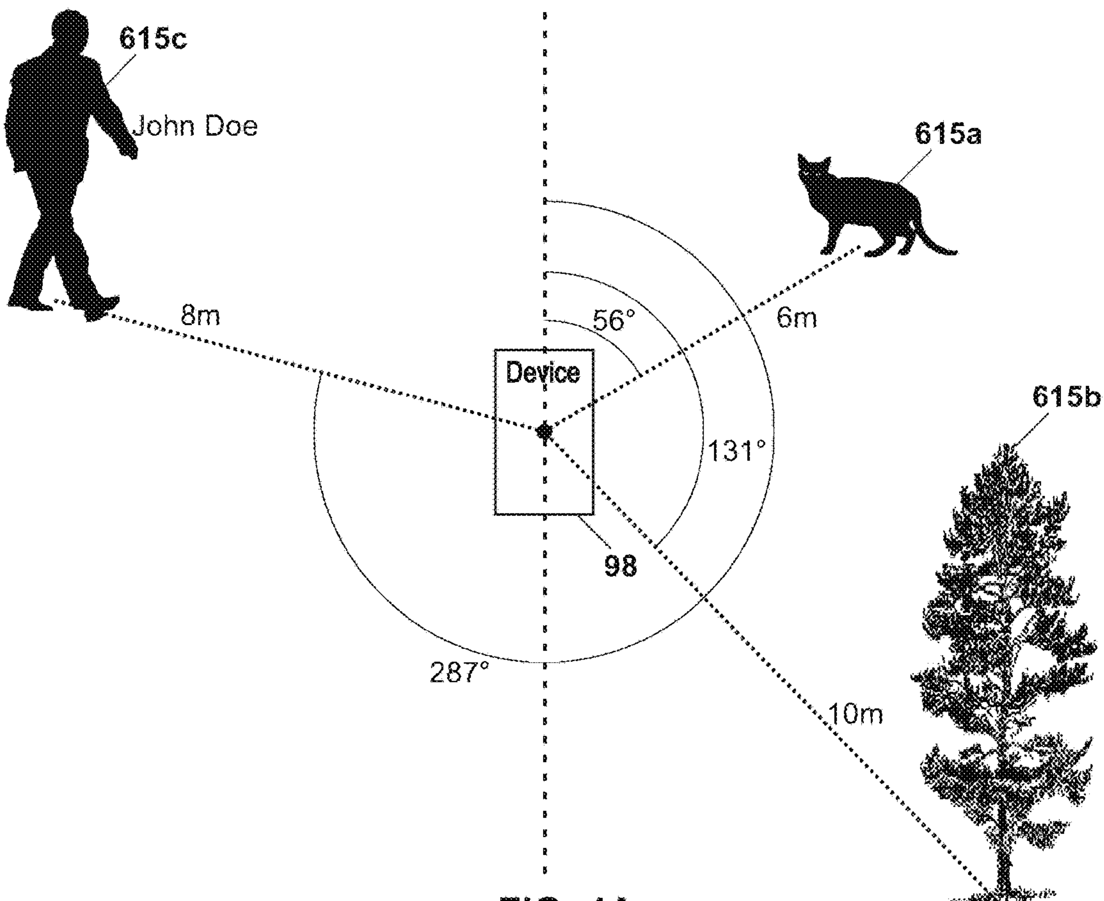
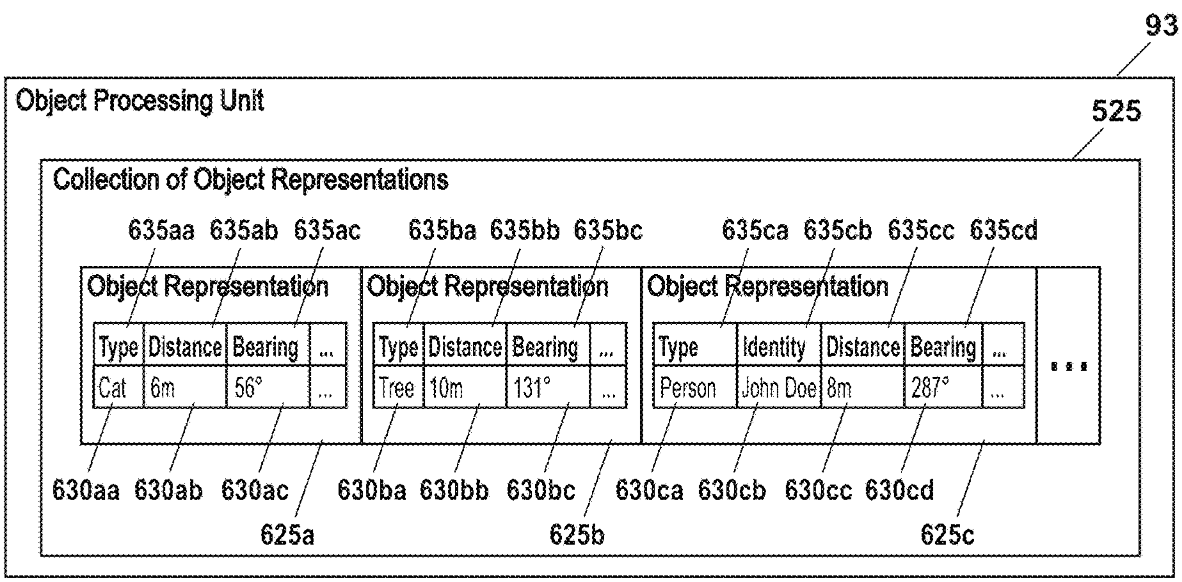


FIG. 3E



615c
John Doe
8m
615a
56°
6m
Device
615b
131°
98
287°
10m

FIG. 4A



93
Object Processing Unit
525
Collection of Object Representations
635aa 635ab 635ac
635ba 635bb 635bc
635ca 635cb 635cc 635cd
Object Representation
Type | Distance | Bearing | ...
Cat | 6m | 56° | ...
Object Representation
Type | Distance | Bearing | ...
Tree | 10m | 131° | ...
Object Representation
Type | Identity | Distance | Bearing | ...
Person | John Doe | 8m | 287° | ...
...
630aa 630ab 630ac
625a
630ba 630bb 630bc
625b
630ca 630cb 630cc 630cd
625c

FIG. 4B

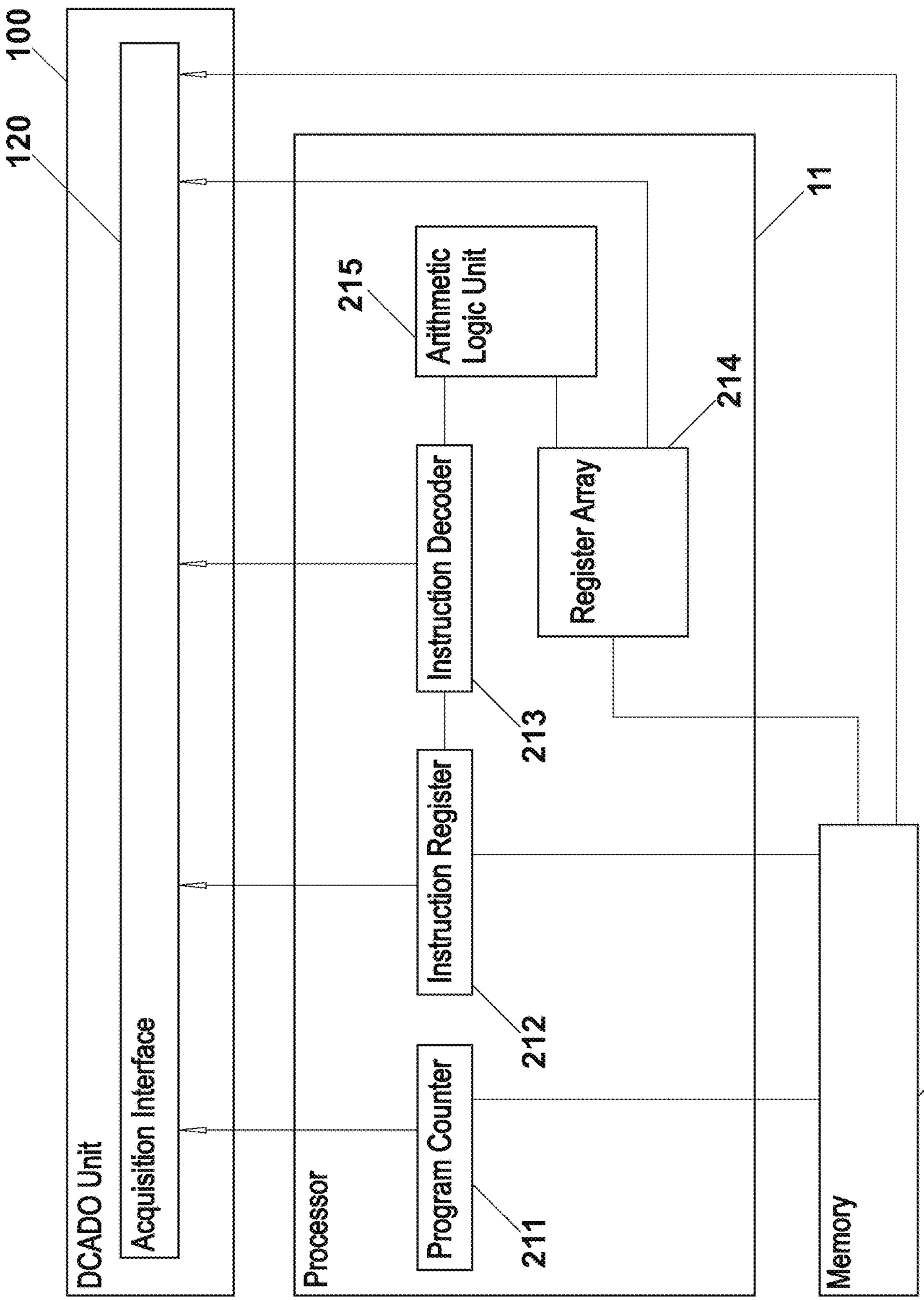

100
120
DCADO Unit
Acquisition Interface
Processor
11
Program Counter
211
Instruction Register
212
Instruction Decoder
213
Register Array
214
Arithmetic Logic Unit
215
Memory
12

FIG. 5

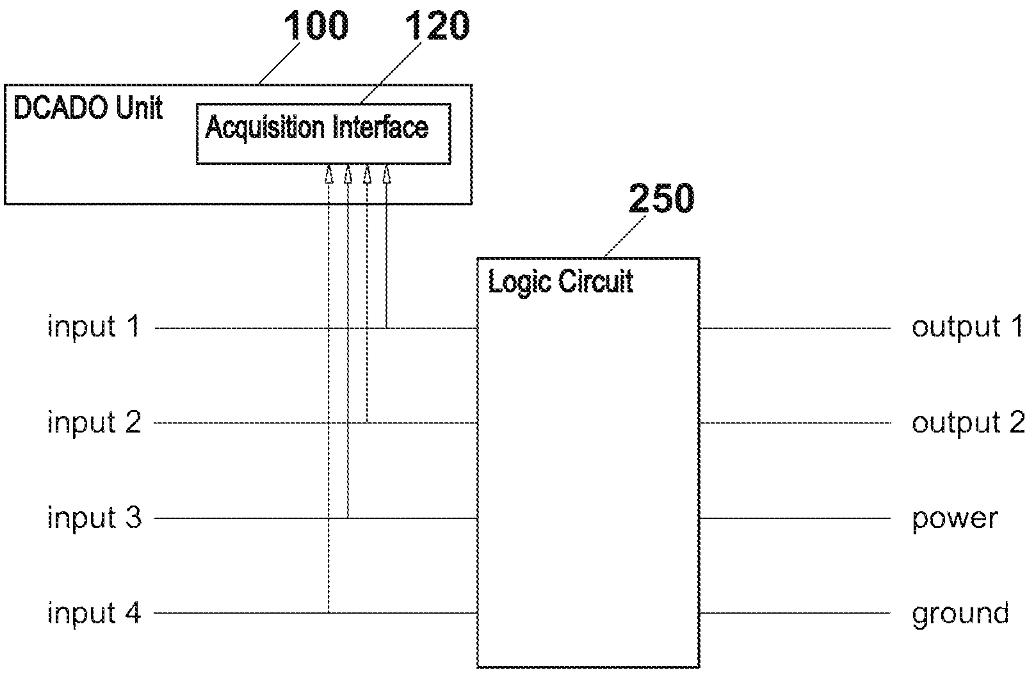

100
120
DCADO Unit
Acquisition Interface
250
Logic Circuit
input 1
input 2
input 3
input 4
output 1
output 2
power
ground

FIG. 6A

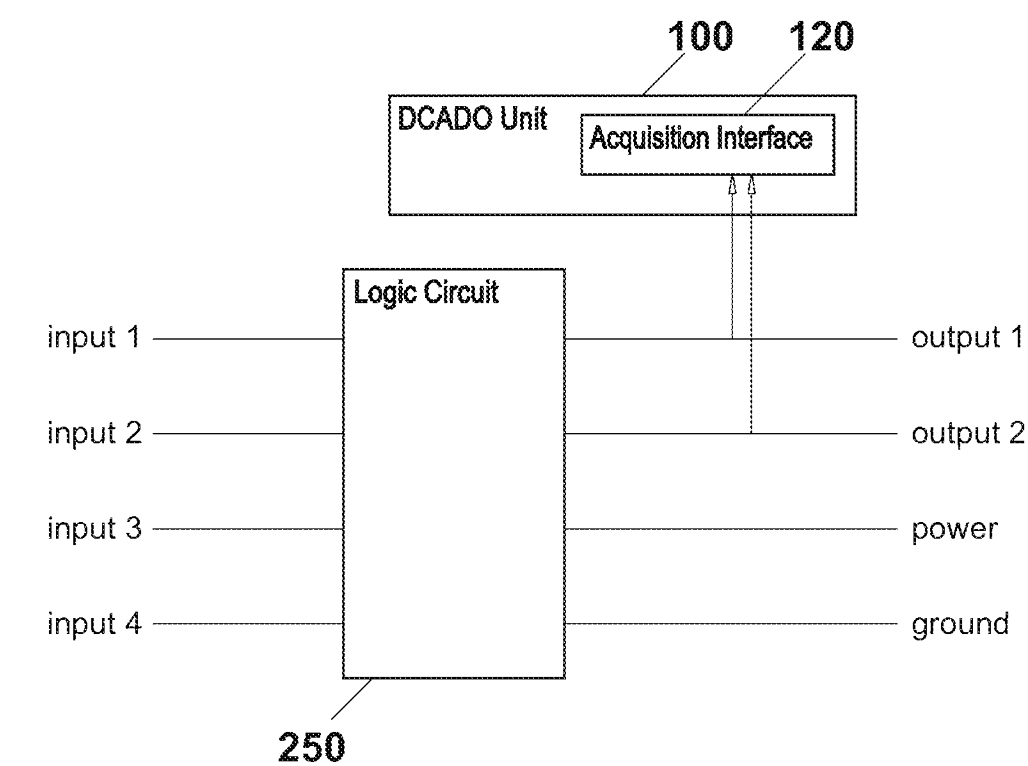

100
120
DCADO Unit
Acquisition Interface
Logic Circuit
input 1
input 2
input 3
input 4
output 1
output 2
power
ground
250

FIG. 6B

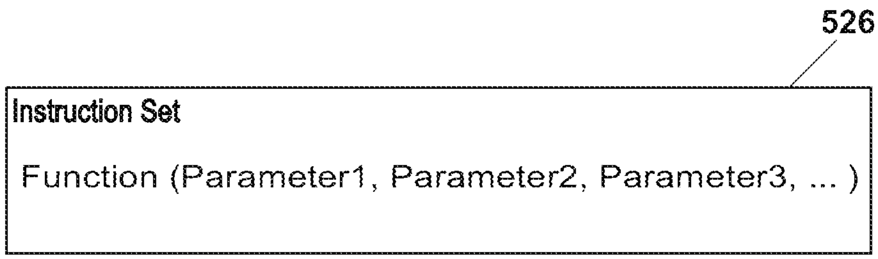


FIG. 7A

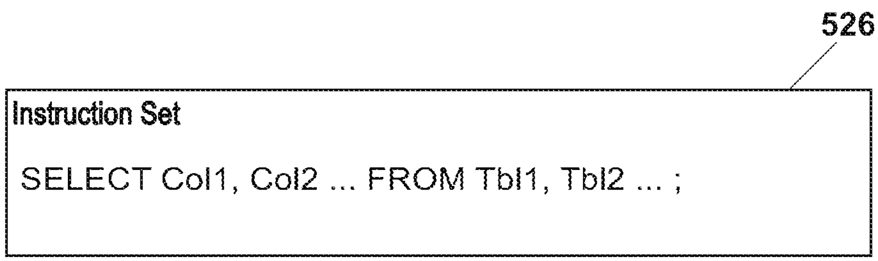


FIG. 7B

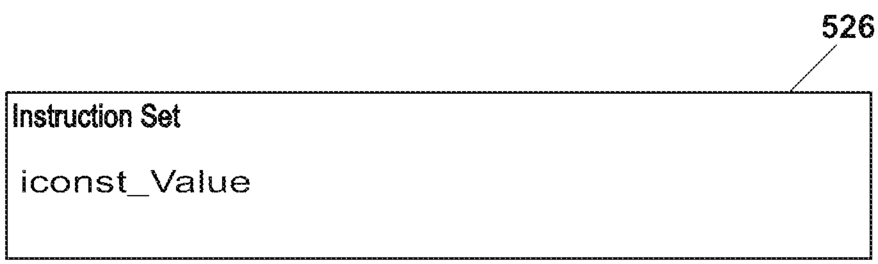


FIG. 7C

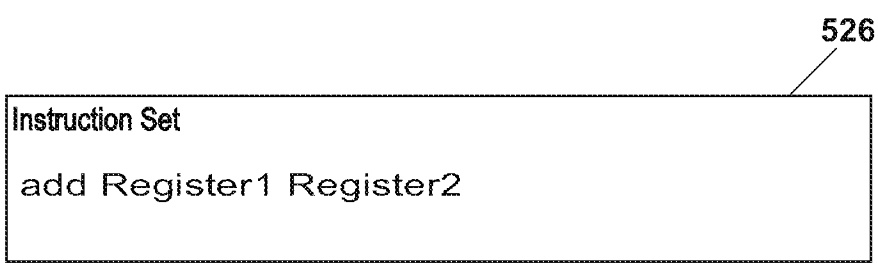


FIG. 7D

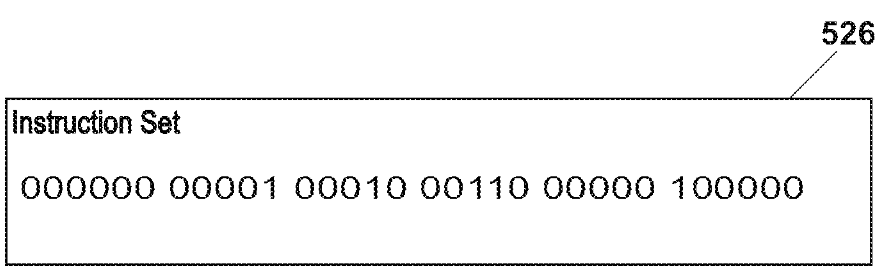


FIG. 7E

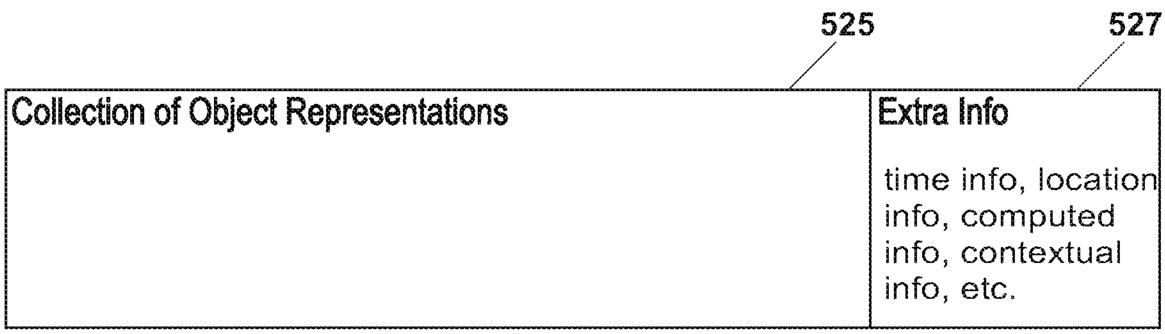

525
527
Collection of Object Representations
Extra Info
time info, location info, computed info, contextual info, etc.

FIG. 8A

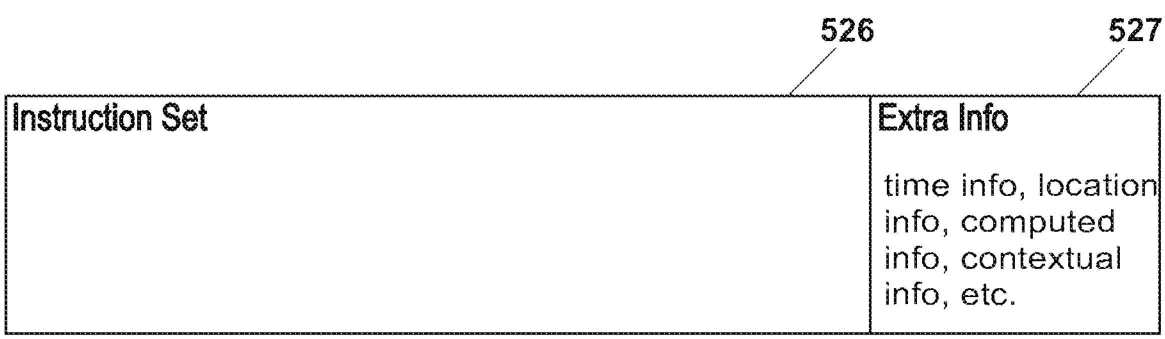

526
527
Instruction Set
Extra Info
time info, location info, computed info, contextual info, etc.

FIG. 8B

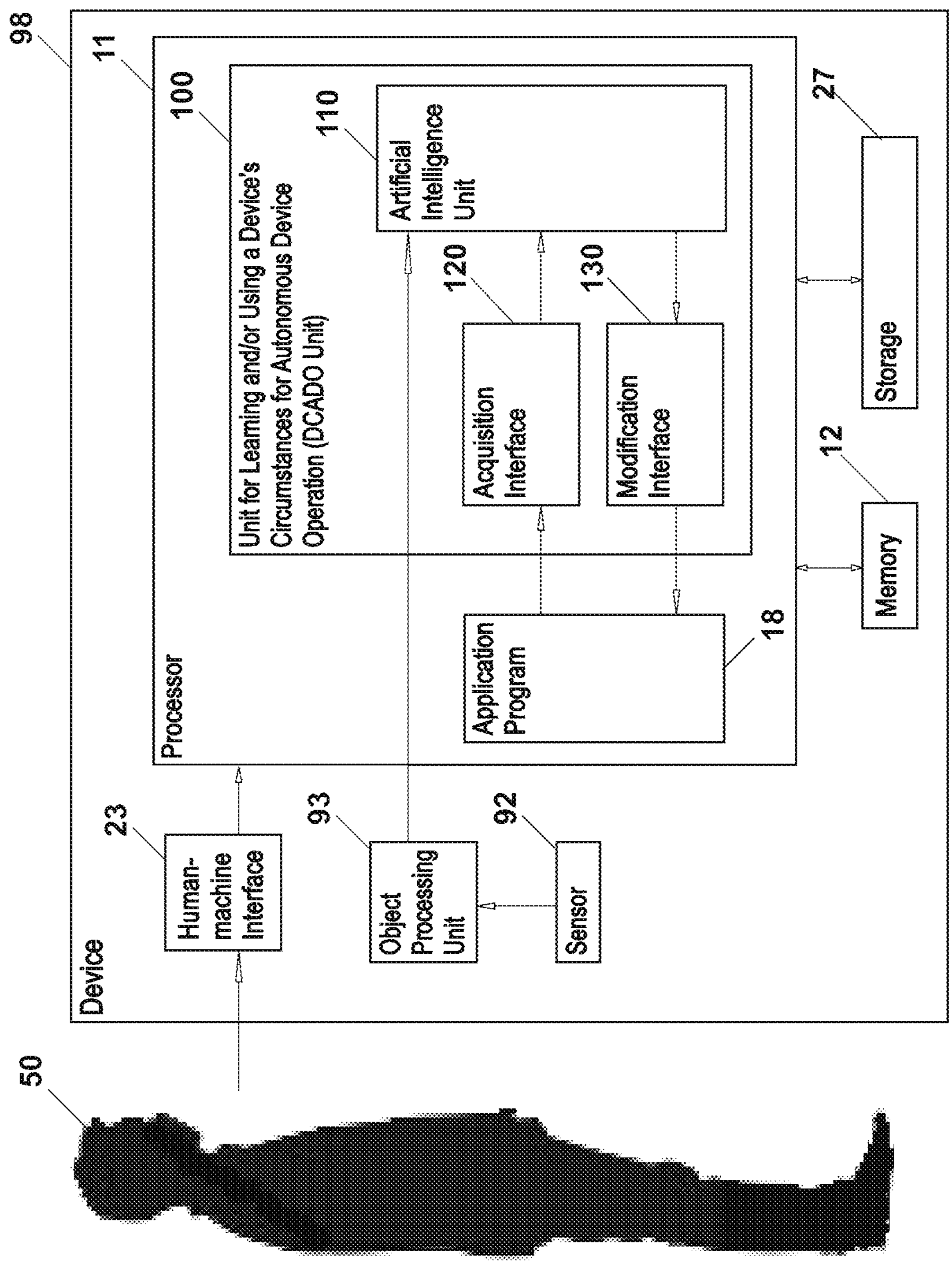

98
Device
11
Processor
100
Unit for Learning and/or Using a Device's Circumstances for Autonomous Device Operation (DCADO Unit)
110
Artificial Intelligence Unit
120
Acquisition Interface
130
Modification Interface
18
Application Program
27
Storage
12
Memory
23
Human-machine Interface
93
Object Processing Unit
92
Sensor
50

FIG. 9

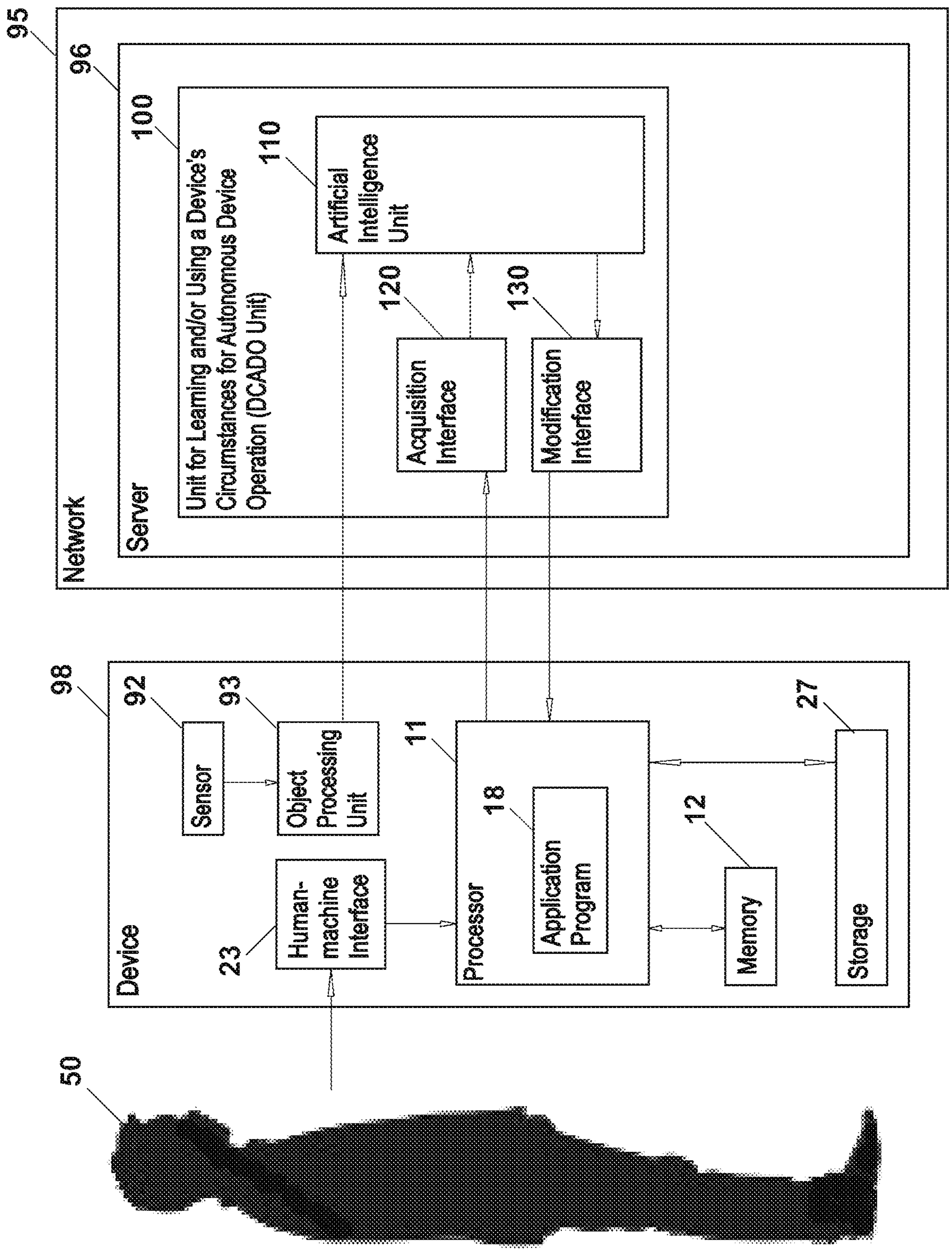

95
Network
96
Server
100
Unit for Learning and/or Using a Device's Circumstances for Autonomous Device Operation (DCADO Unit)
110
Artificial Intelligence Unit
120
Acquisition Interface
130
Modification Interface
98
Device
92
Sensor
93
Object Processing Unit
23
Human-machine Interface
11
Processor
18
Application Program
12
Memory
27
Storage
50

FIG. 10

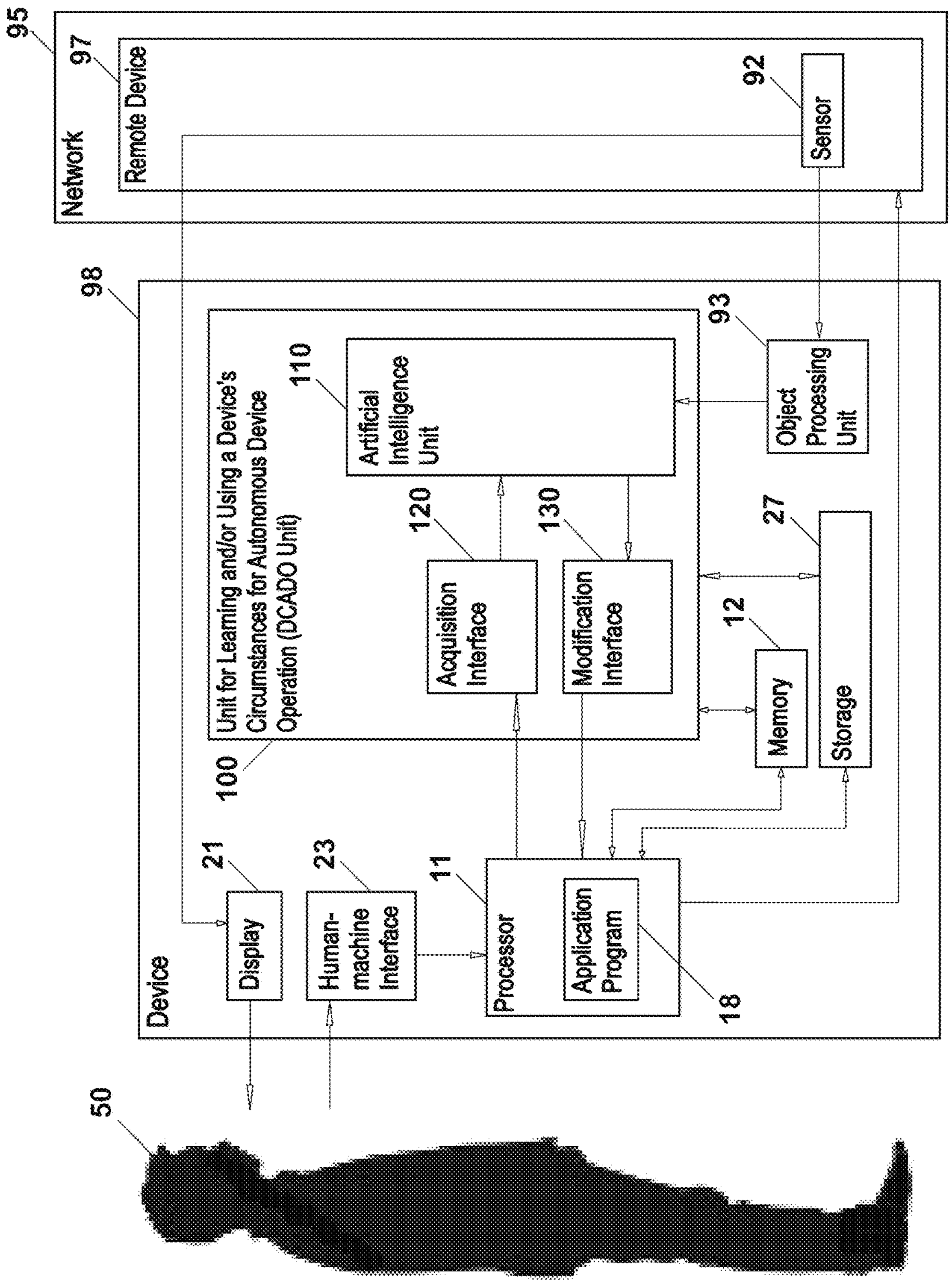

95
Network
97
Remote Device
92
Sensor
98
Device
93
Object Processing Unit
110
Artificial Intelligence Unit
Unit for Learning and/or Using a Device's Circumstances for Autonomous Device Operation (DCADO Unit)
100
120
Acquisition Interface
130
Modification Interface
27
Storage
12
Memory
21
Display
23
Human-machine Interface
11
Processor
Application Program
18
50

FIG. 11

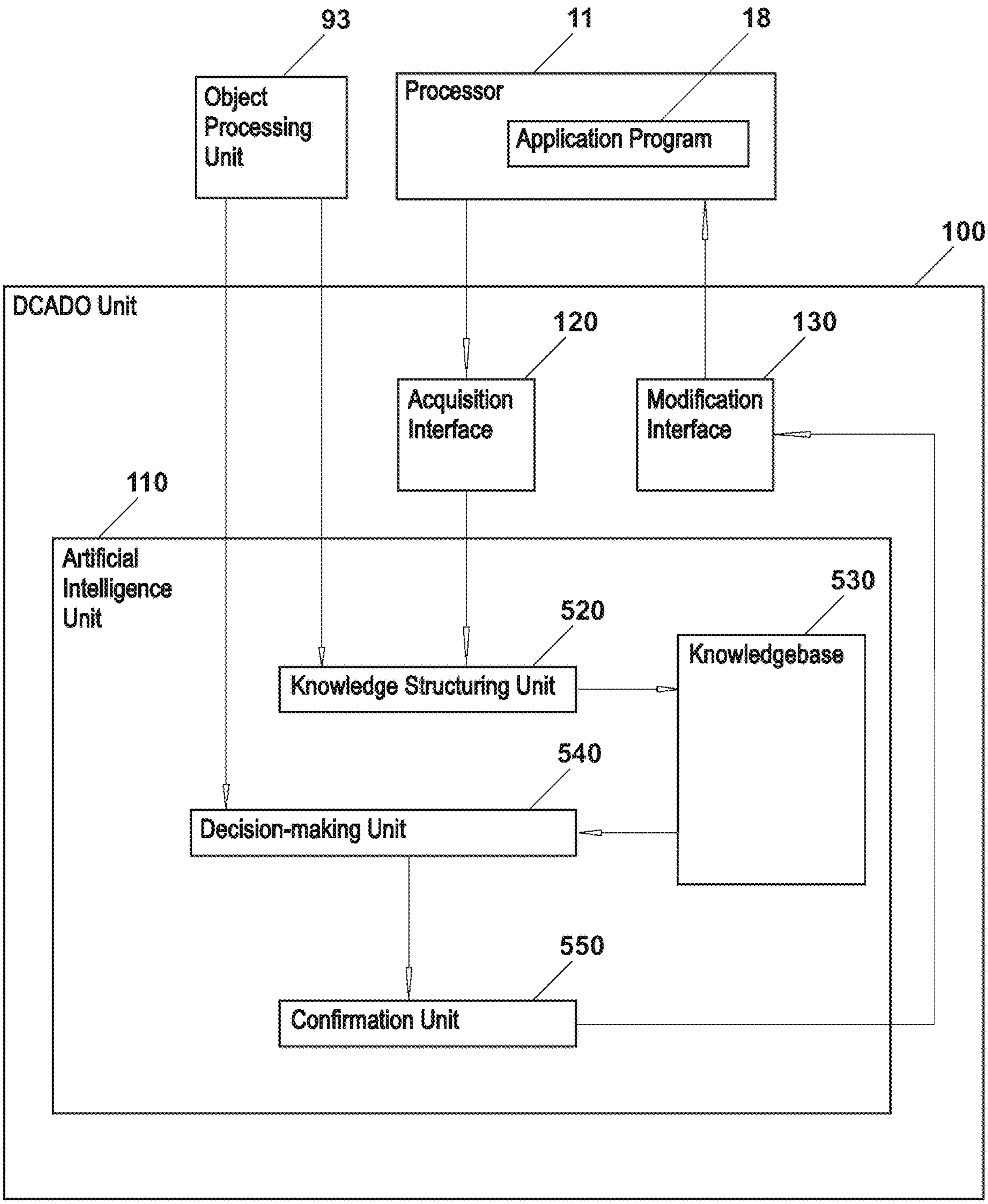
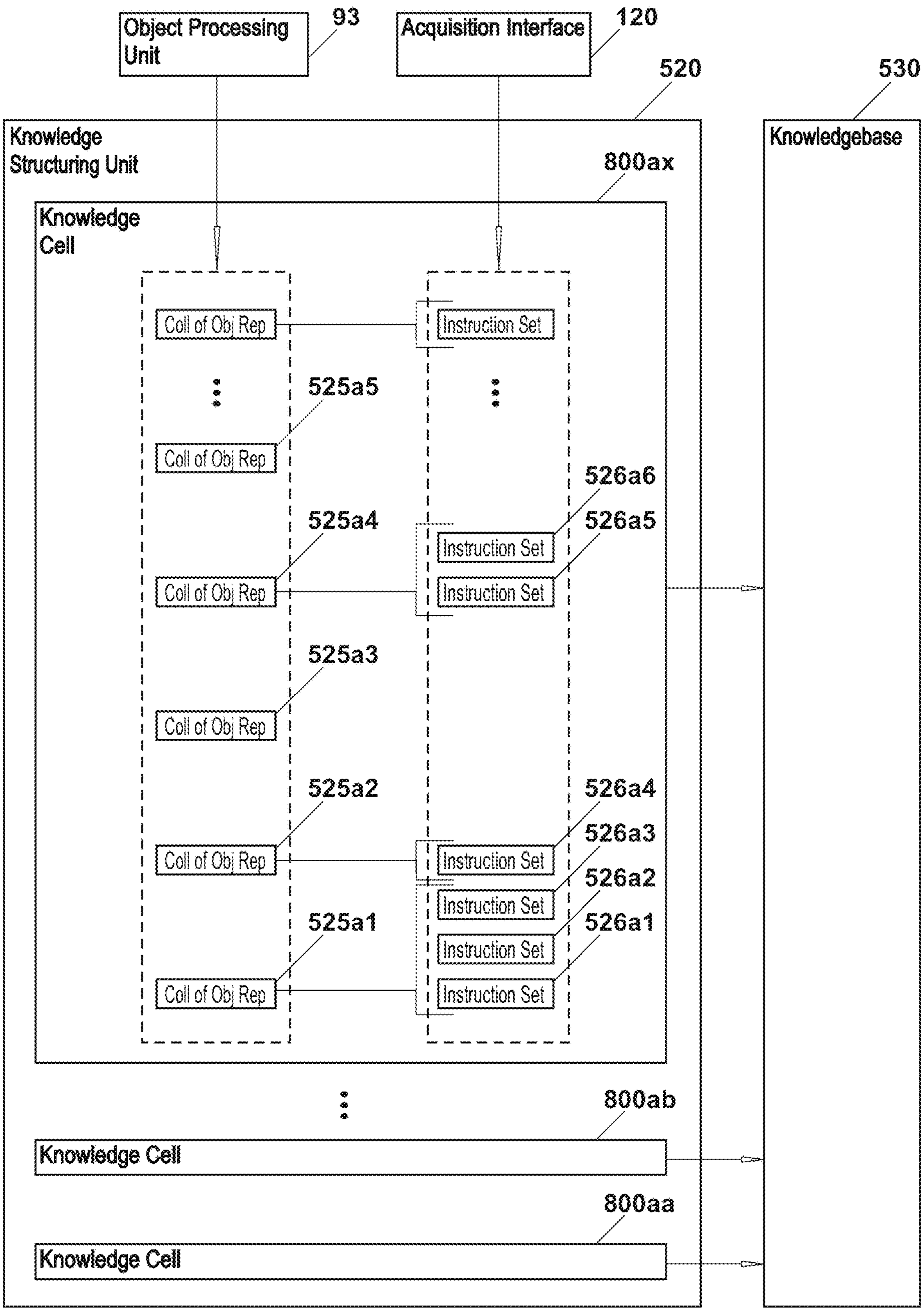
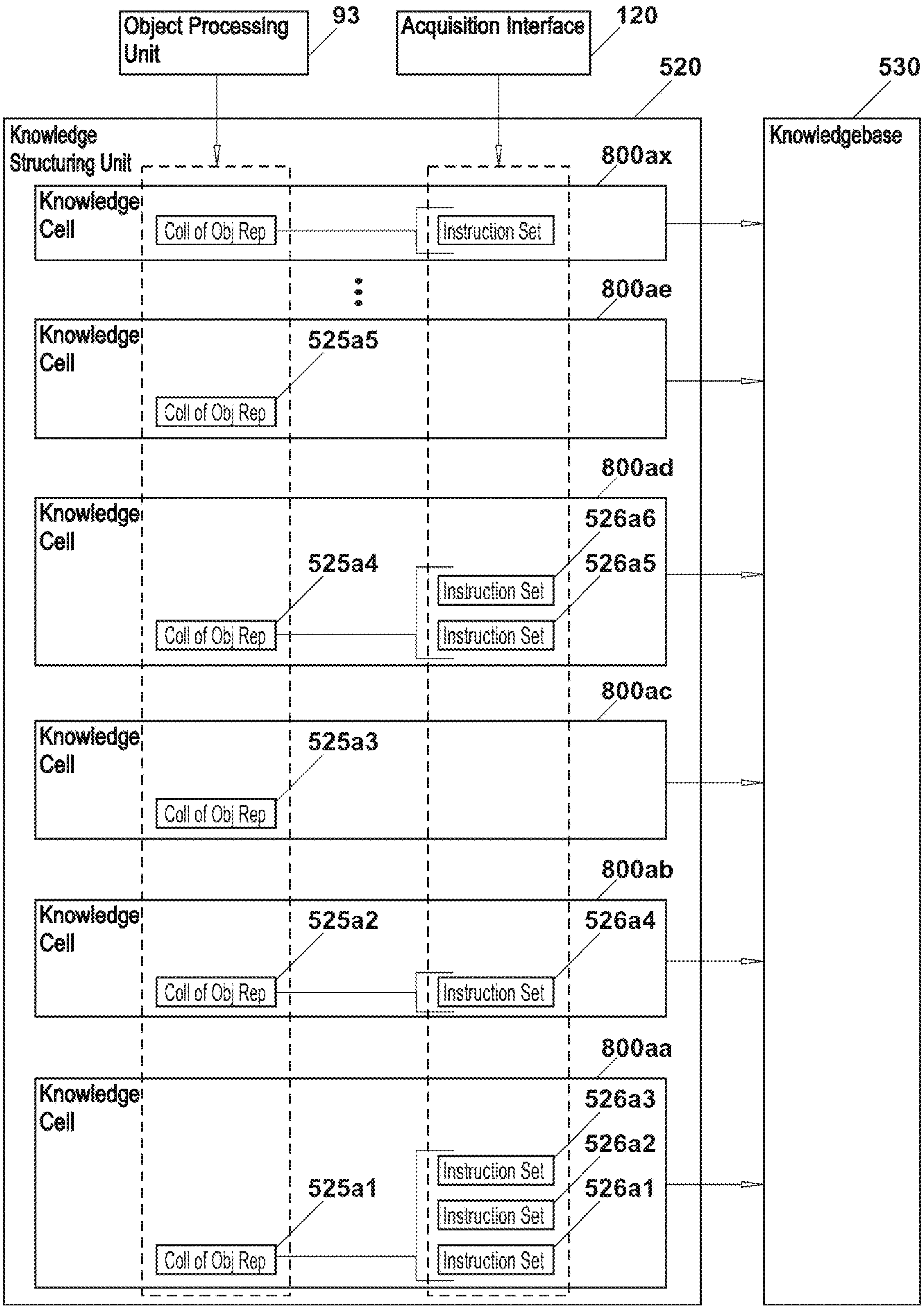

93
11
18
Object Processing Unit
Processor
Application Program
100
DCADO Unit
120
130
Acquisition Interface
Modification Interface
110
Artificial Intelligence Unit
530
520
Knowledgebase
Knowledge Structuring Unit
540
Decision-making Unit
550
Confirmation Unit

FIG. 12



Object Processing Unit
93
Acquisition Interface
120
520
530
Knowledge Structuring Unit
Knowledgebase
800ax
Knowledge Cell
Coll of Obj Rep
Instruction Set
525a5
Coll of Obj Rep
526a6
525a4
526a5
Instruction Set
Coll of Obj Rep
Instruction Set
525a3
Coll of Obj Rep
525a2
526a4
526a3
Coll of Obj Rep
Instruction Set
526a2
Instruction Set
525a1
526a1
Instruction Set
Coll of Obj Rep
Instruction Set
800ab
Knowledge Cell
800aa
Knowledge Cell

FIG. 13



Object Processing Unit
93
Acquisition Interface
120
520
530
Knowledge Structuring Unit
Knowledgebase
800ax
Knowledge Cell
Coll of Obj Rep
Instruction Set
800ae
Knowledge Cell
525a5
Coll of Obj Rep
800ad
Knowledge Cell
526a6
525a4
526a5
Instruction Set
Coll of Obj Rep
Instruction Set
800ac
Knowledge Cell
525a3
Coll of Obj Rep
800ab
Knowledge Cell
525a2
526a4
Coll of Obj Rep
Instruction Set
800aa
Knowledge Cell
526a3
526a2
525a1
526a1
Instruction Set
Instruction Set
Coll of Obj Rep
Instruction Set

FIG. 14

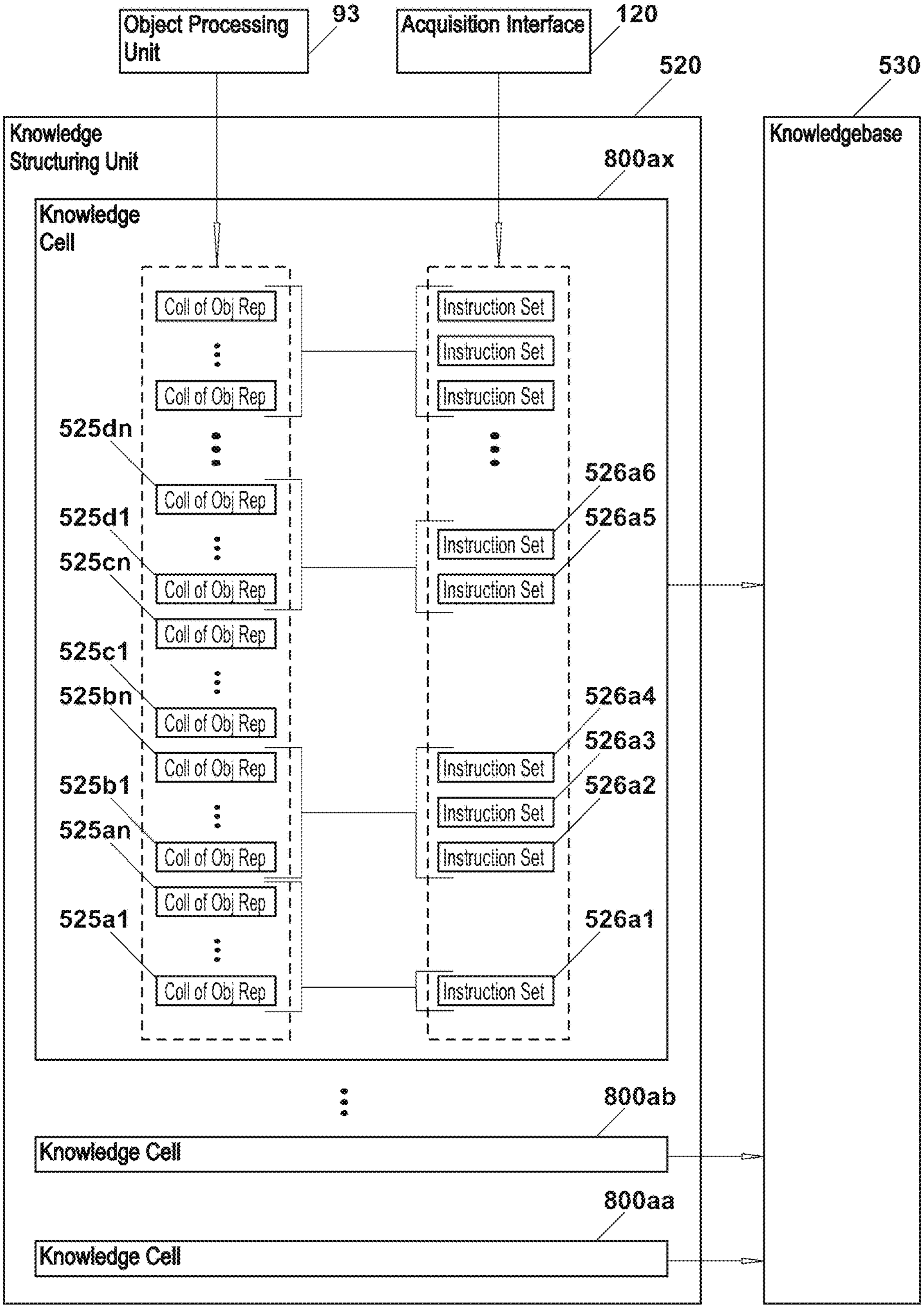
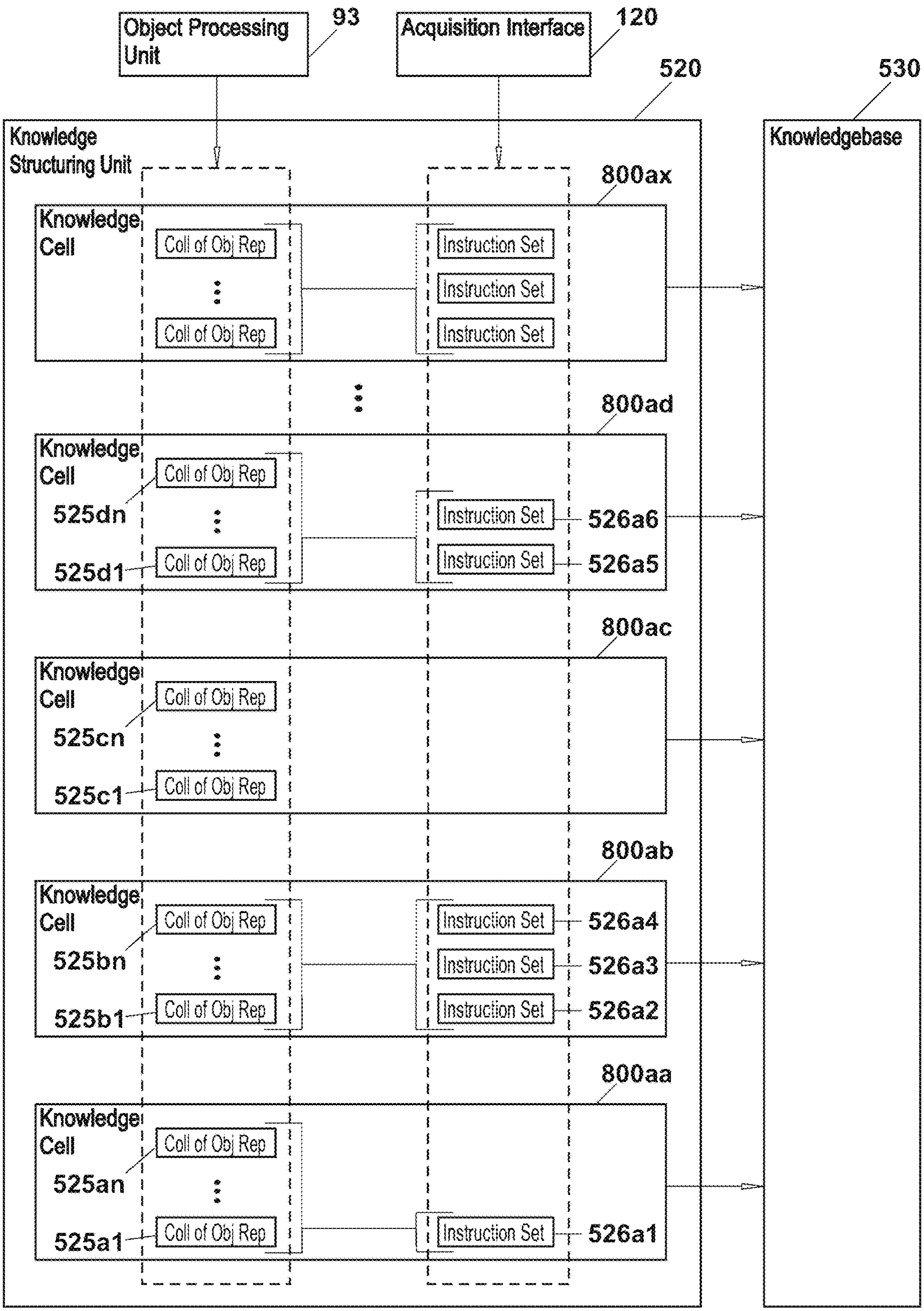

Object Processing Unit
93
Acquisition Interface
120
520
530
Knowledge Structuring Unit
800ax
Knowledgebase
Knowledge Cell
Coll of Obj Rep
Instruction Set
525dn
525d1
525cn
525c1
525bn
525b1
525an
525a1
526a6
526a5
526a4
526a3
526a2
526a1
800ab
Knowledge Cell
800aa
Knowledge Cell

FIG. 15



Object Processing Unit
93
Acquisition Interface
120
520
530
Knowledge Structuring Unit
Knowledgebase
800ax
Knowledge Cell
Coll of Obj Rep
Coll of Obj Rep
Instruction Set
Instruction Set
Instruction Set
800ad
Knowledge Cell
525dn
525d1
Coll of Obj Rep
Coll of Obj Rep
Instruction Set
Instruction Set
526a6
526a5
800ac
Knowledge Cell
525cn
525c1
Coll of Obj Rep
Coll of Obj Rep
800ab
Knowledge Cell
525bn
525b1
Coll of Obj Rep
Coll of Obj Rep
Instruction Set
Instruction Set
Instruction Set
526a4
526a3
526a2
800aa
Knowledge Cell
525an
525a1
Coll of Obj Rep
Coll of Obj Rep
Instruction Set
526a1

FIG. 16

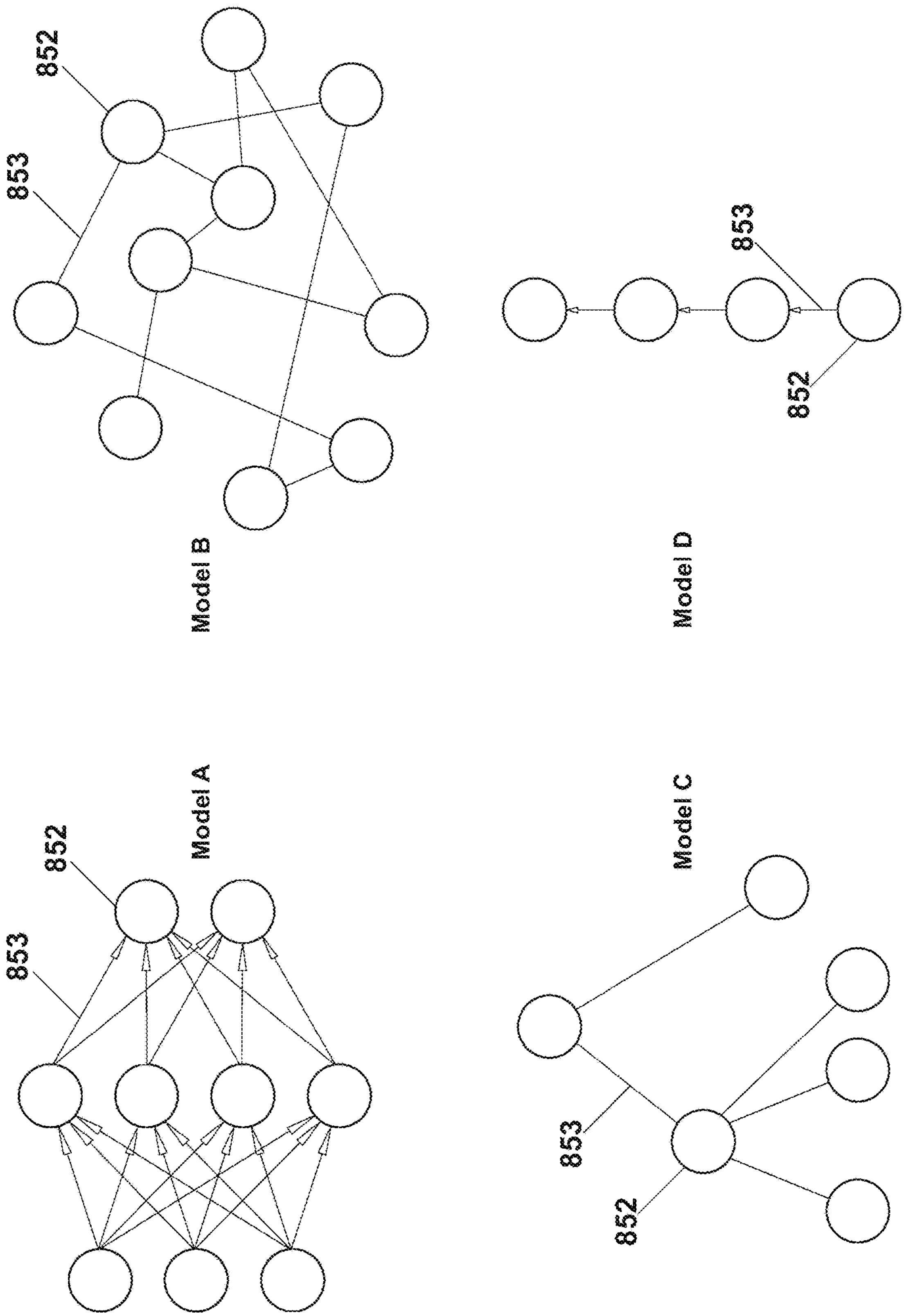


FIG. 17

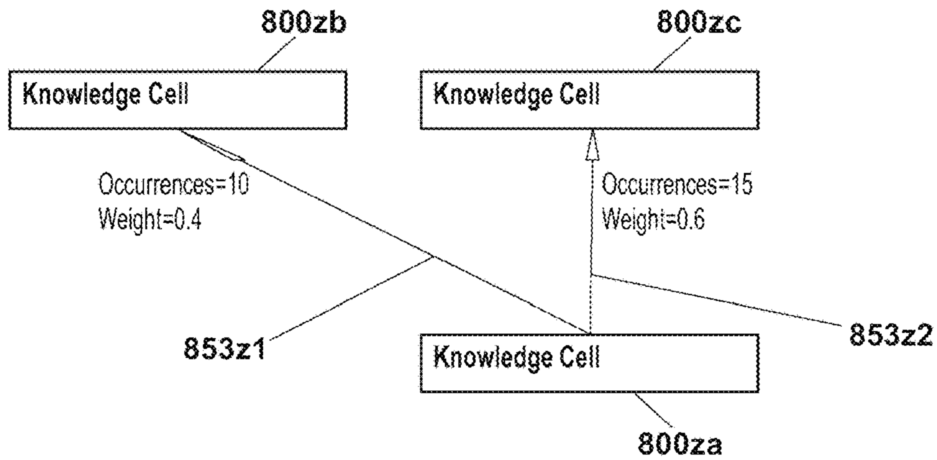


FIG. 18A

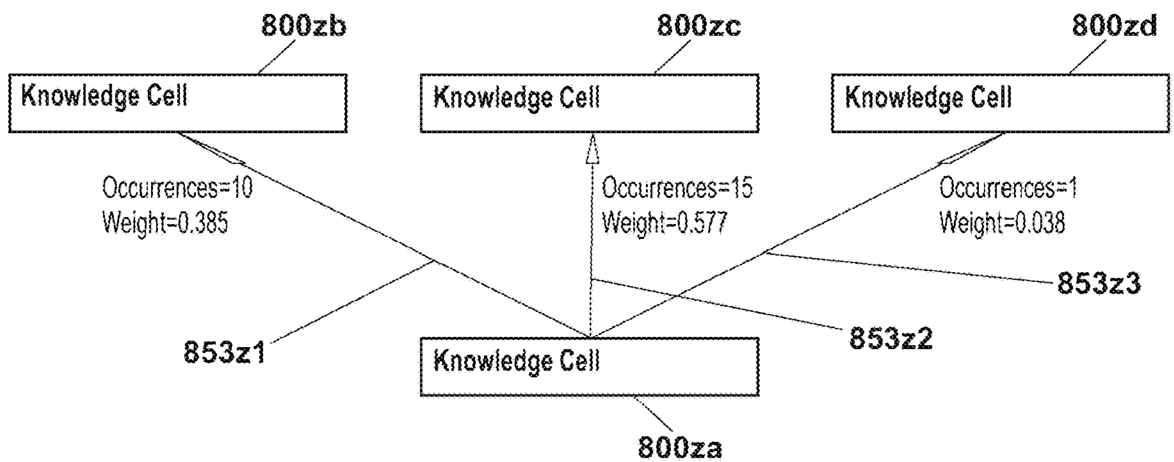


FIG. 18B

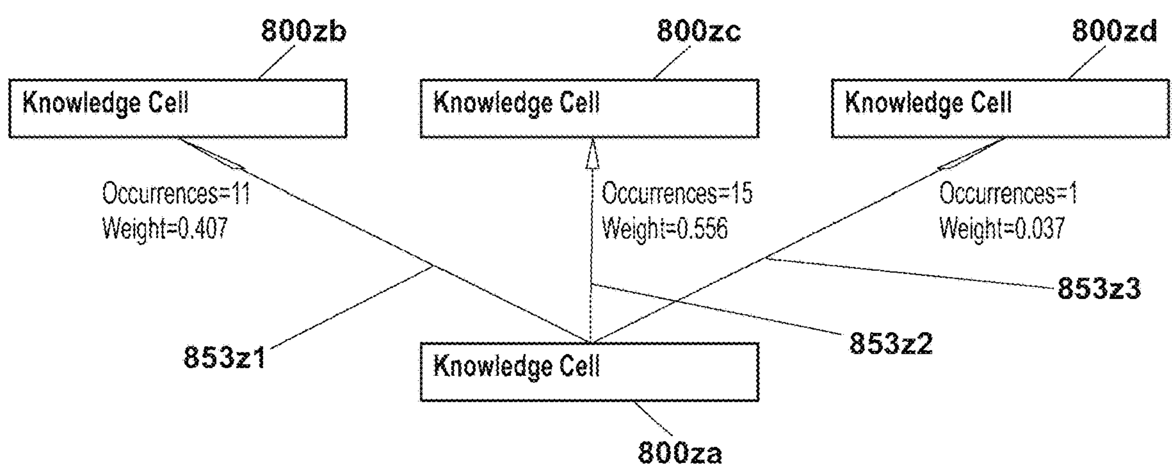


FIG. 18C

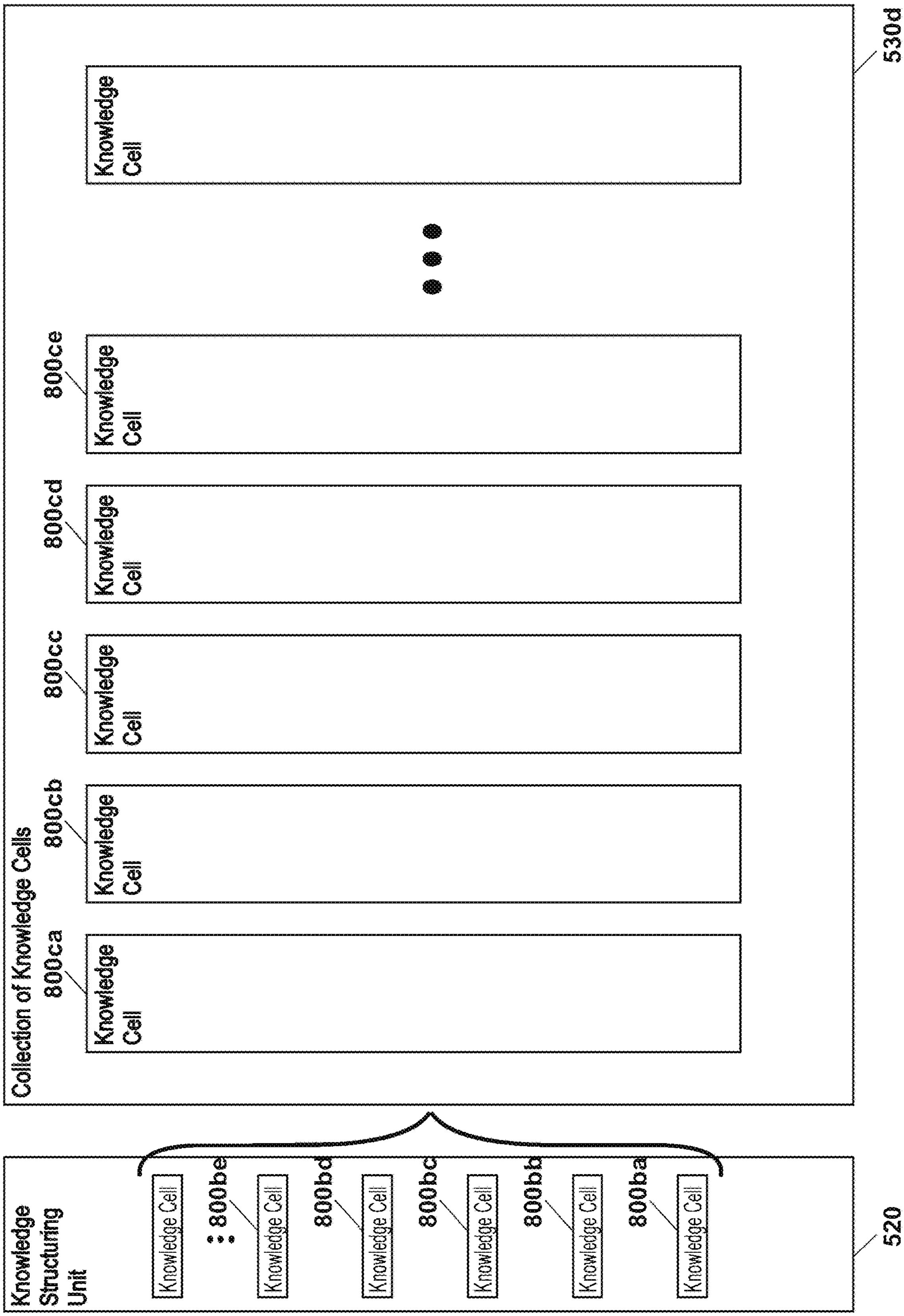

Collection of Knowledge Cells
800ca
Knowledge Cell
800cb
Knowledge Cell
800cc
Knowledge Cell
800cd
Knowledge Cell
800ce
Knowledge Cell
Knowledge Cell
530d
Knowledge Structuring Unit
Knowledge Cell
800be
Knowledge Cell
800bd
Knowledge Cell
800bc
Knowledge Cell
800bb
Knowledge Cell
800ba
Knowledge Cell
520

FIG. 19

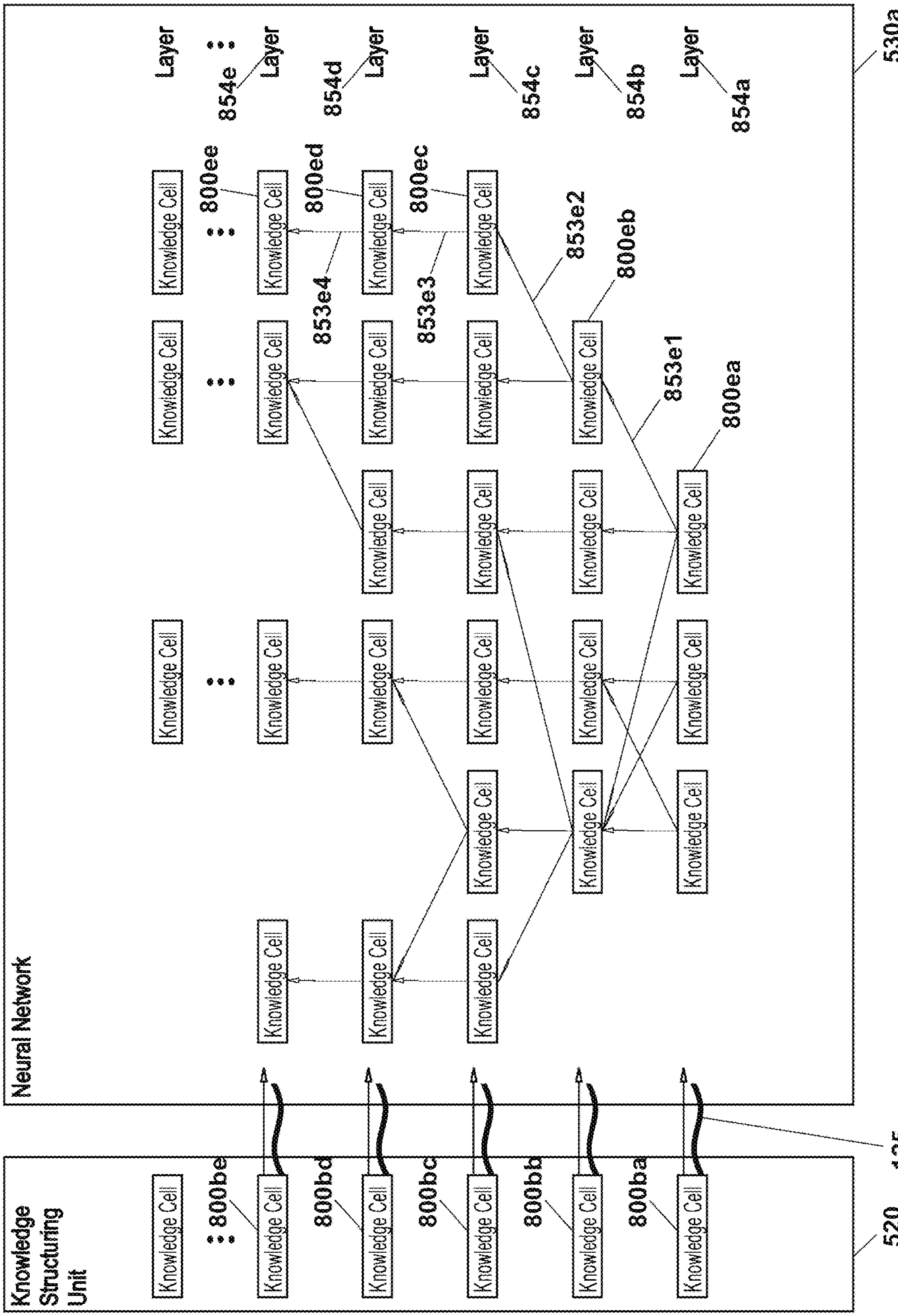

Knowledge Structuring Unit
Neural Network
Knowledge Cell
Layer
520
530a
125
800ba
800bb
800bc
800bd
800be
800ea
800eb
800ec
800ed
800ee
853e1
853e2
853e3
853e4
854a
854b
854c
854d
854e

FIG. 20

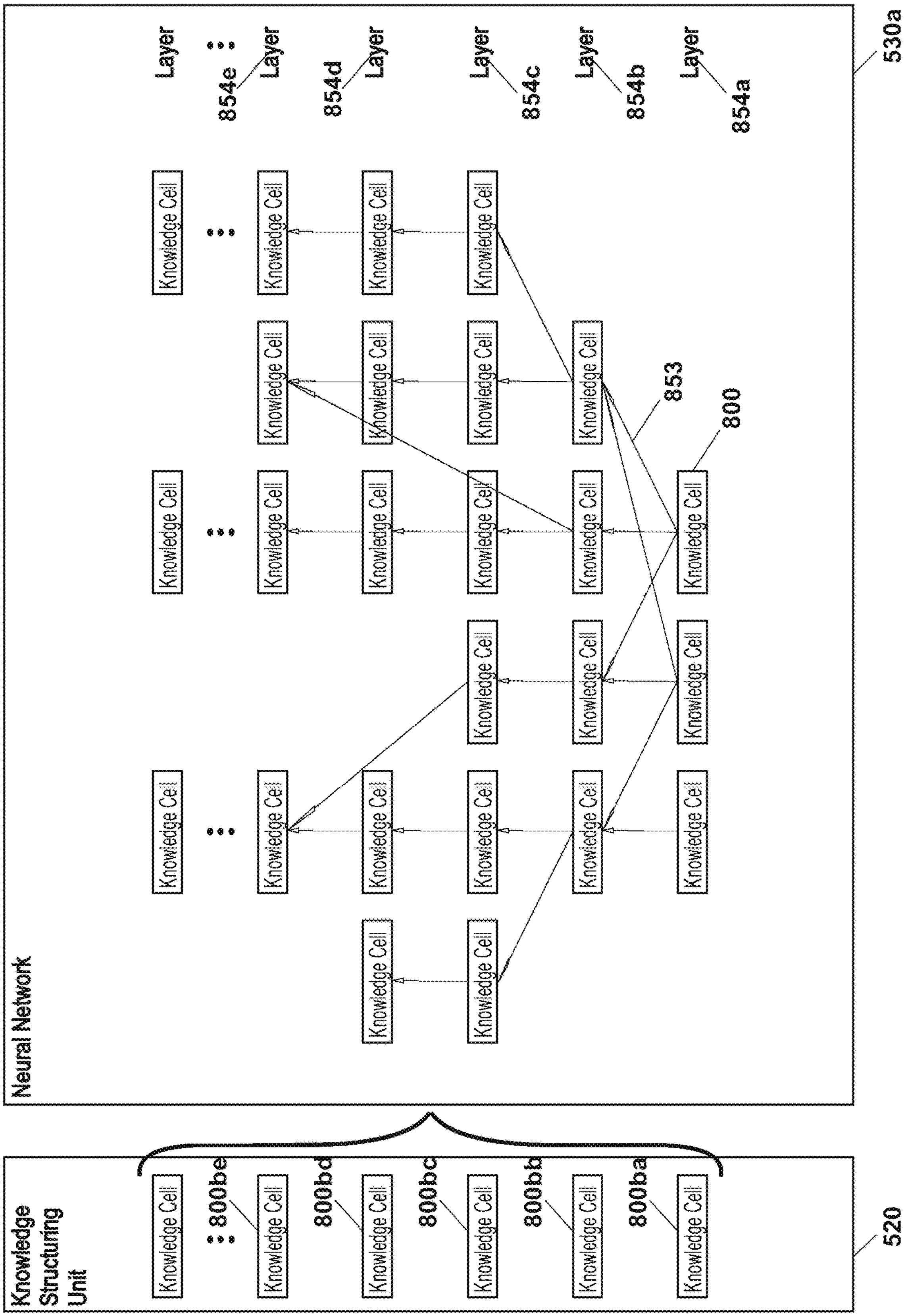

Neural Network
Layer
Knowledge Cell
854a
854b
854c
854d
854e
853
800
530a
Knowledge Structuring Unit
800ba
800bb
800bc
800bd
800be
520

FIG. 21

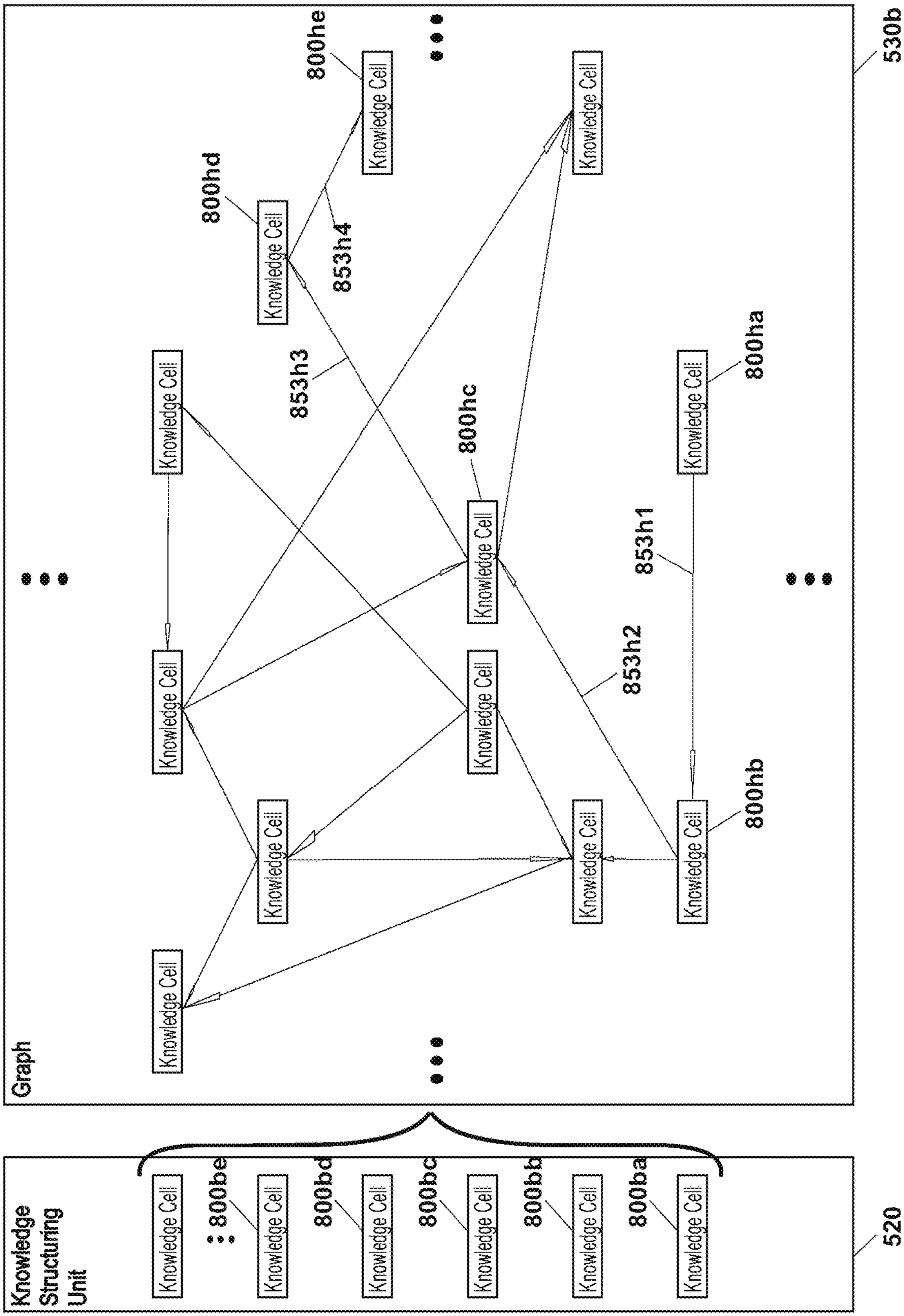

Graph
Knowledge Cell
800ha
800hb
800hc
800hd
800he
853h1
853h2
853h3
853h4
530b
Knowledge Structuring Unit
800ba
800bb
800bc
800bd
800be
520

FIG. 22

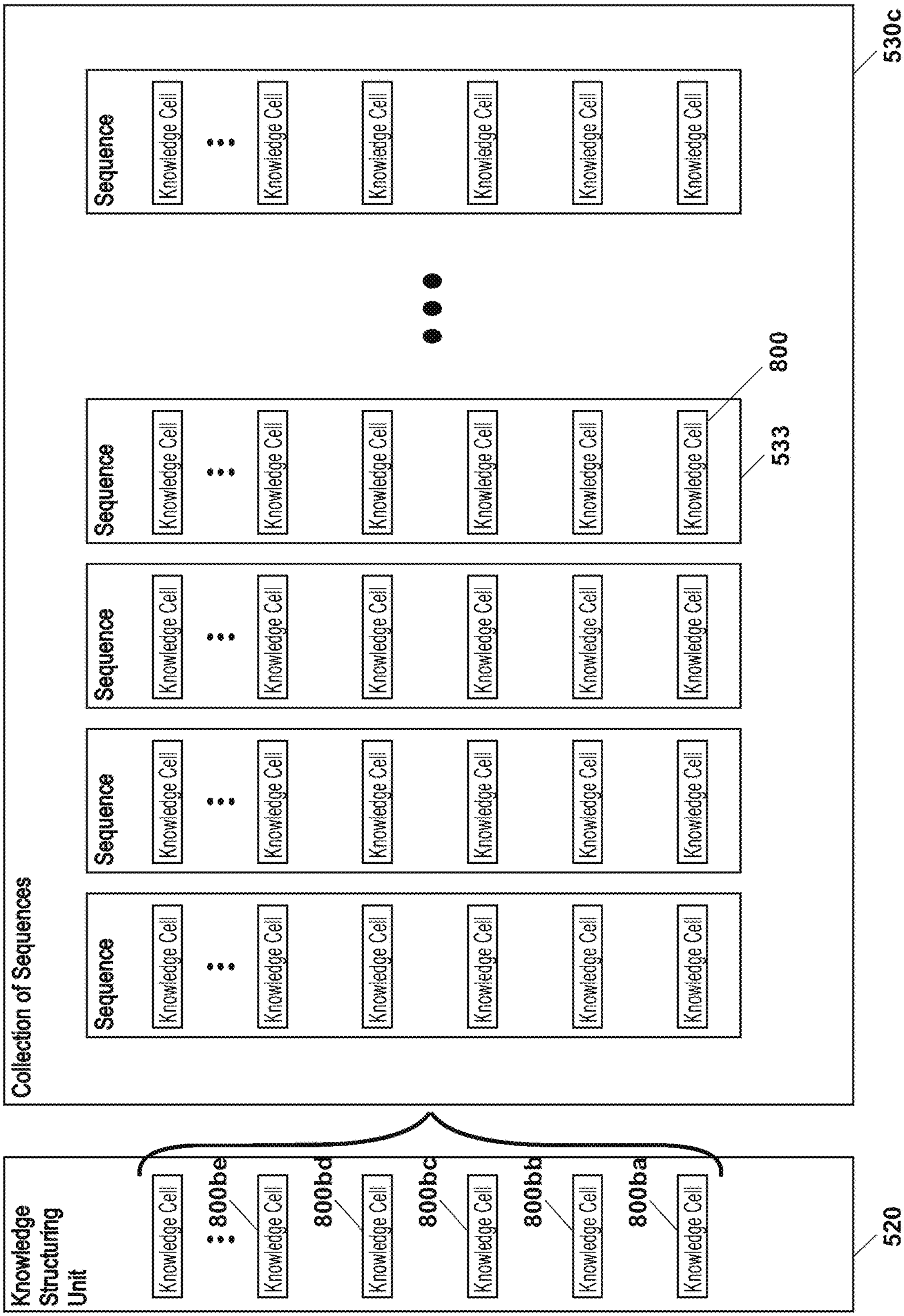
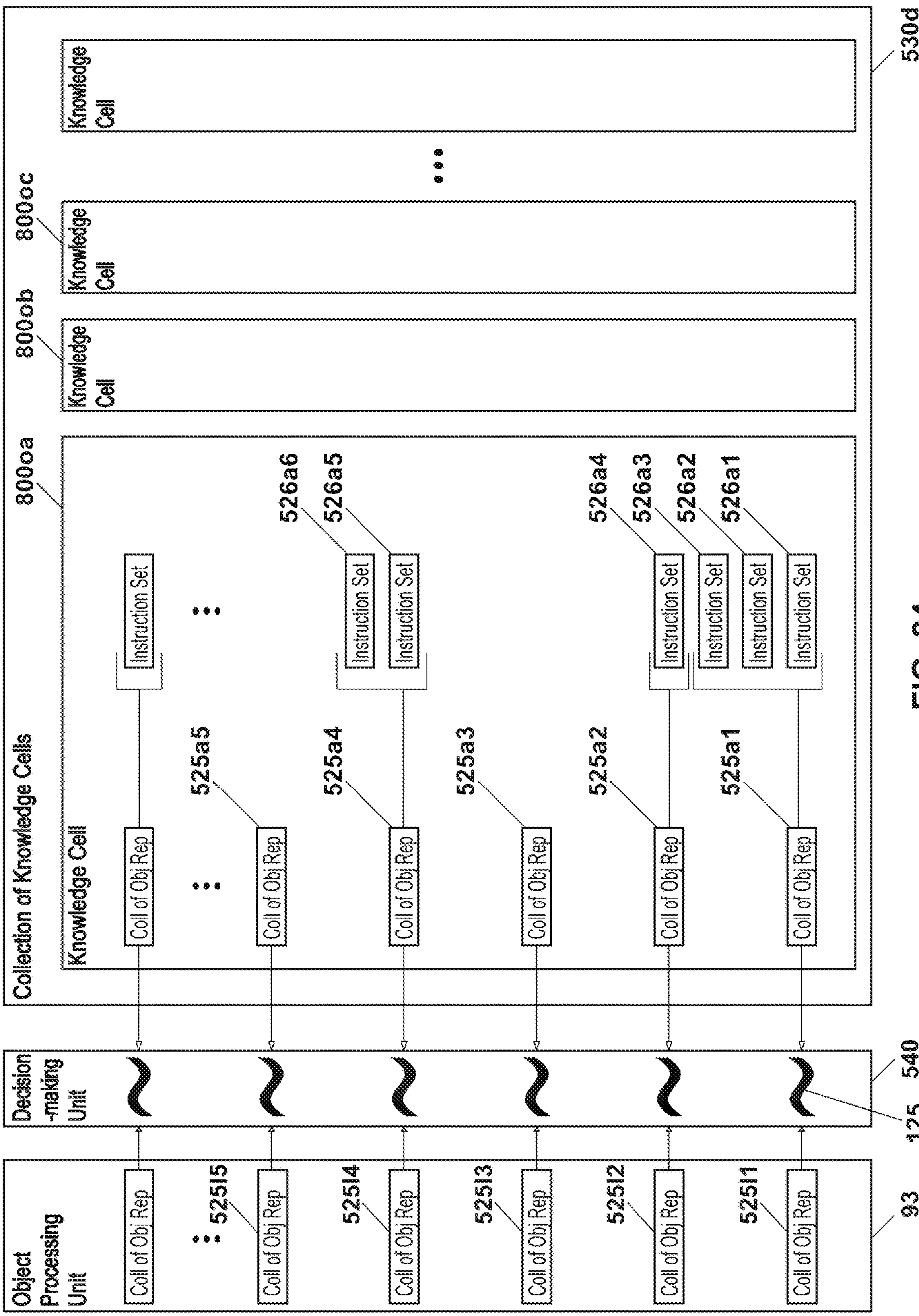

Collection of Sequences
Sequence
Knowledge Cell
530c
533
800
Knowledge Structuring Unit
800ba
800bb
800bc
800bd
800be
520

FIG. 23



Object Processing Unit
Coll of Obj Rep
525l1
525l2
525l3
525l4
525l5
93
Decision -making Unit
125
540
Collection of Knowledge Cells
Knowledge Cell
800oa
525a1
525a2
525a3
525a4
525a5
Instruction Set
526a1
526a2
526a3
526a4
526a5
526a6
Knowledge Cell
800ob
800oc
530d

FIG. 24

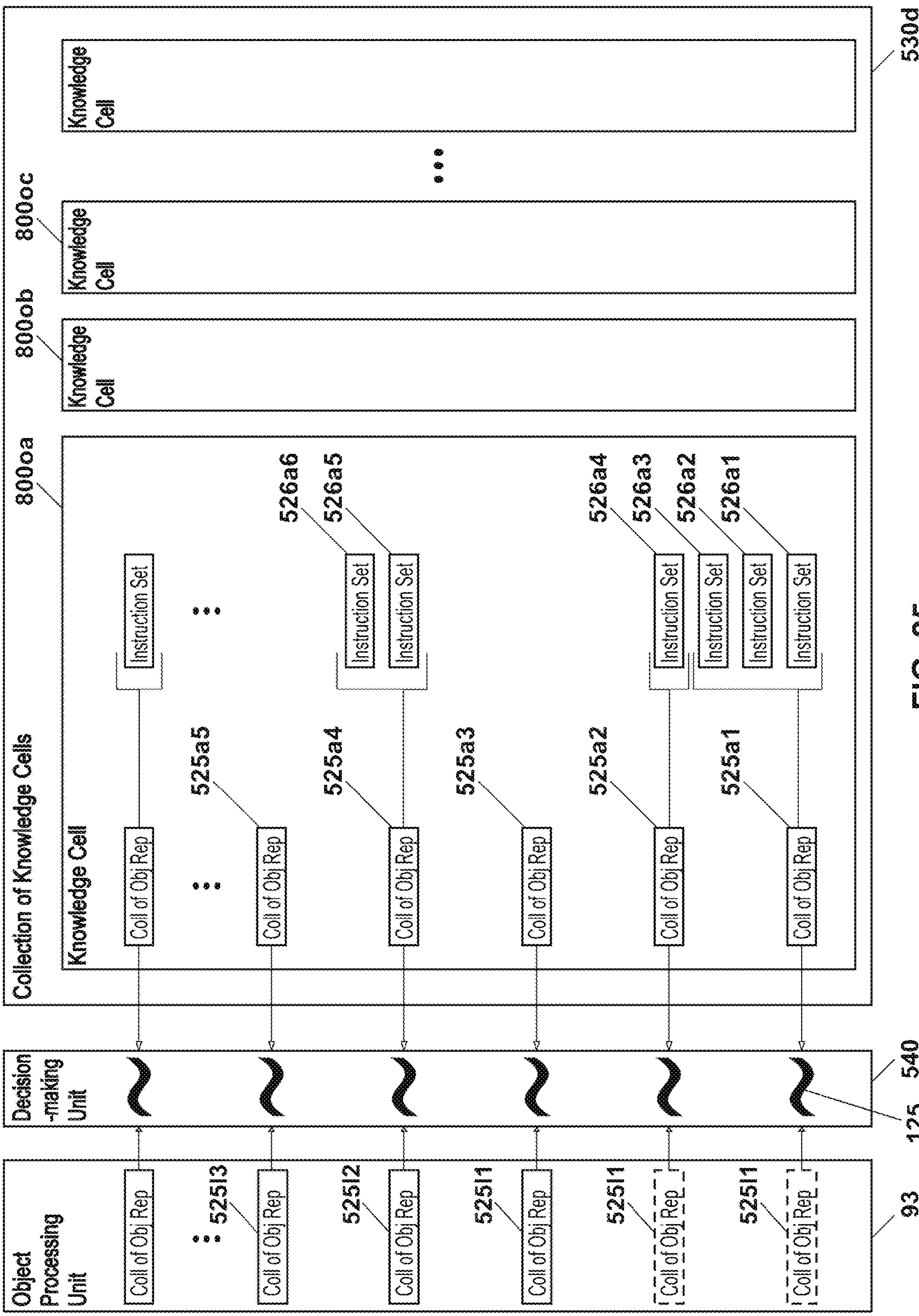

Collection of Knowledge Cells
Knowledge Cell
Knowledge Cell
Knowledge Cell
Knowledge Cell
800oa
800ob
800oc
530d
Coll of Obj Rep
Instruction Set
525a1
525a2
525a3
525a4
525a5
526a1
526a2
526a3
526a4
526a5
526a6
Decision -making Unit
540
125
Object Processing Unit
93
525l1
525l2
525l3

FIG. 25

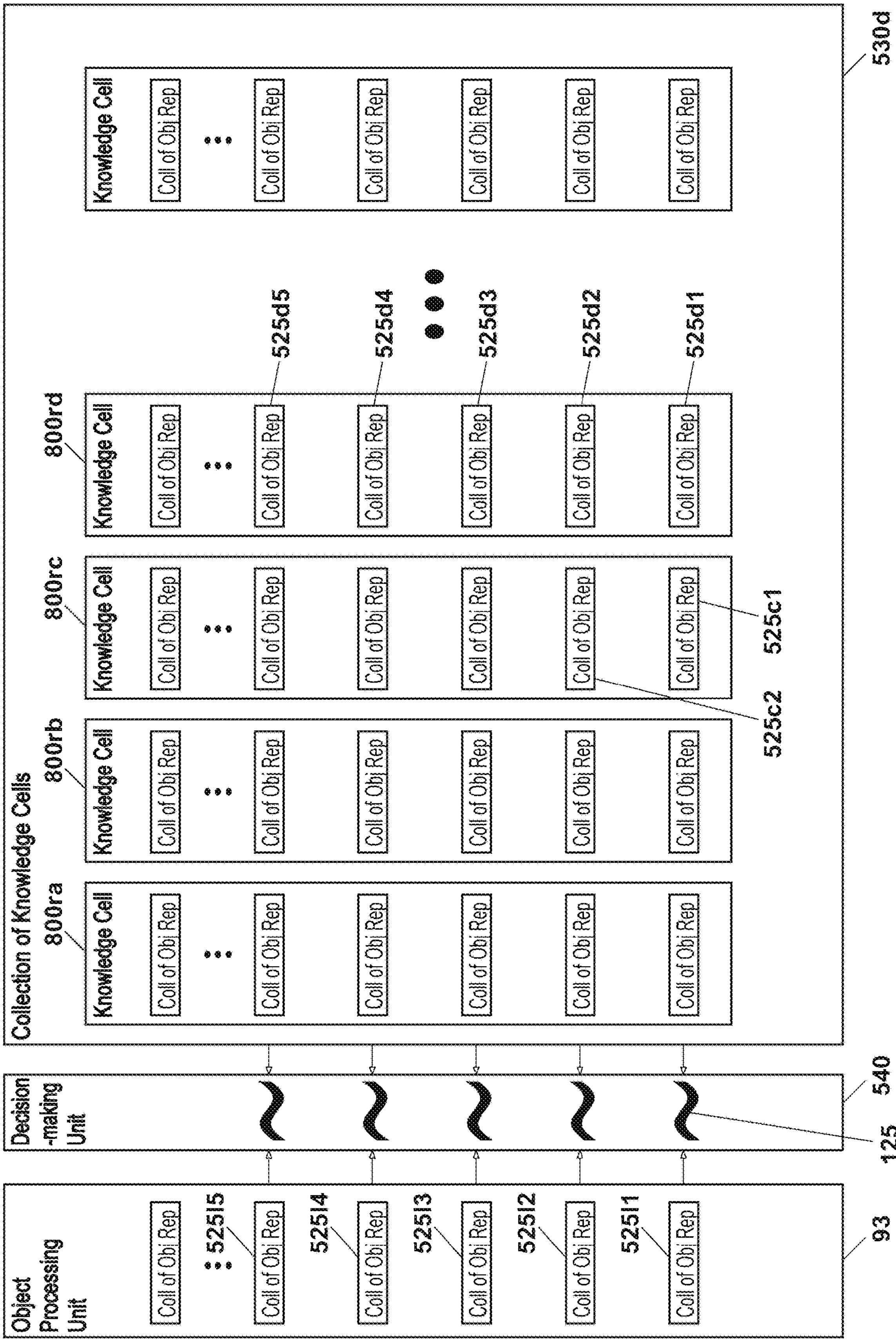

Object Processing Unit
Coll of Obj Rep
525l5
525l4
525l3
525l2
525l1
93
Decision -making Unit
125
540
Collection of Knowledge Cells
800ra
800rb
800rc
800rd
Knowledge Cell
525c1
525c2
525d1
525d2
525d3
525d4
525d5
530d

FIG. 26

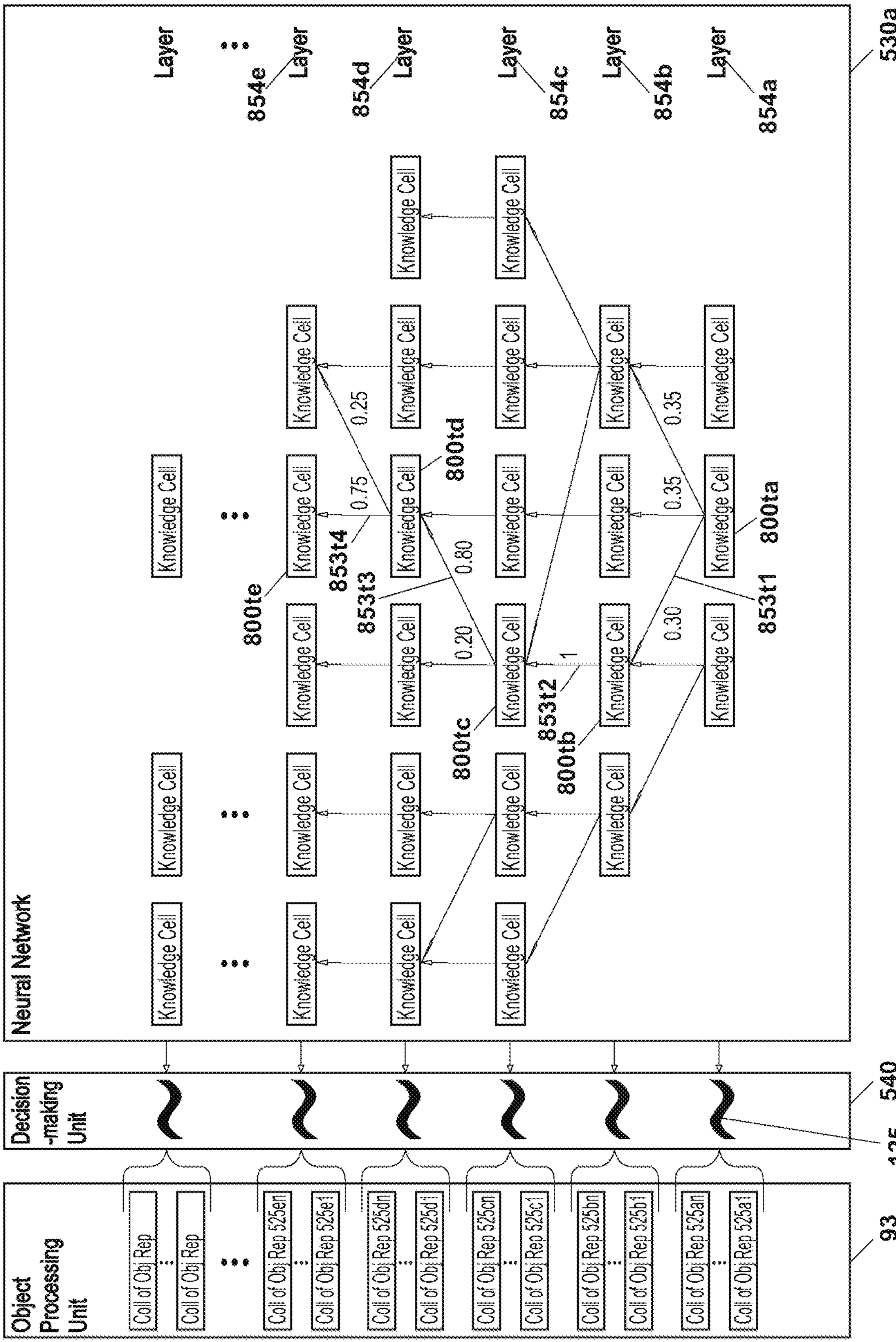

Neural Network
Layer
854e
854d
854c
854b
854a
530a
Knowledge Cell
0.25
0.75
0.35
0.35
0.80
0.20
0.30
1
800td
800te
800ta
800tc
800tb
853t4
853t3
853t2
853t1
Decision -making Unit
540
125
Object Processing Unit
Coll of Obj Rep
Coll of Obj Rep 525em
Coll of Obj Rep 525e1
Coll of Obj Rep 525dn
Coll of Obj Rep 525d1
Coll of Obj Rep 525cn
Coll of Obj Rep 525c1
Coll of Obj Rep 525bn
Coll of Obj Rep 525b1
Coll of Obj Rep 525an
Coll of Obj Rep 525a1
93

FIG. 27

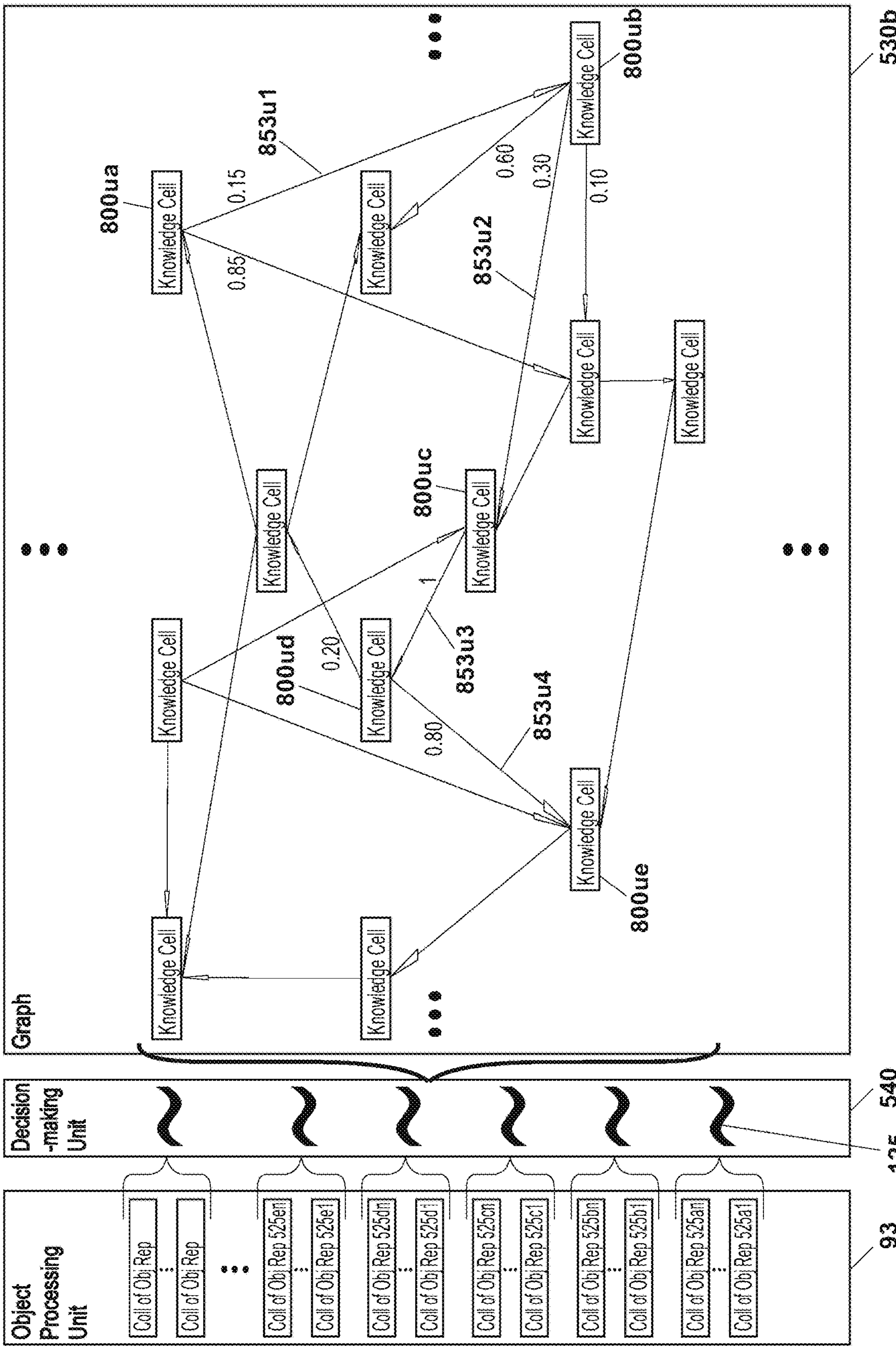
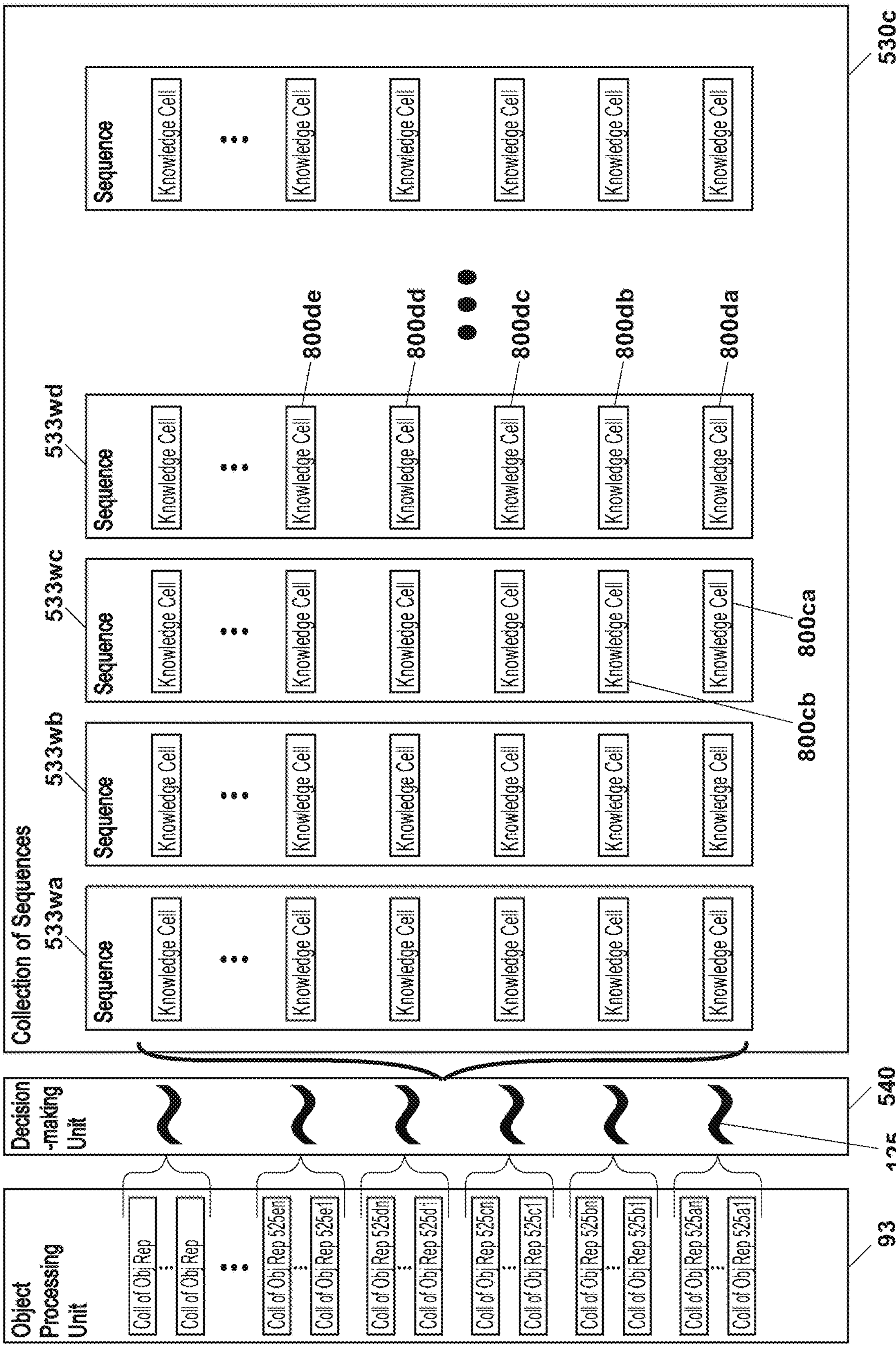

Graph
Knowledge Cell
800ua
800ub
800uc
800ud
800ue
853u1
853u2
853u3
853u4
0.15
0.85
0.60
0.30
0.10
0.20
0.80
1
530b
Decision -making Unit
540
125
Object Processing Unit
Coll of Obj Rep
Coll of Obj Rep 525en
Coll of Obj Rep 525e1
Coll of Obj Rep 525dn
Coll of Obj Rep 525d1
Coll of Obj Rep 525cn
Coll of Obj Rep 525c1
Coll of Obj Rep 525bn
Coll of Obj Rep 525b1
Coll of Obj Rep 525an
Coll of Obj Rep 525a1
93

FIG. 28



Object Processing Unit
Coll of Obj Rep
Coll of Obj Rep
Coll of Obj Rep 525en
Coll of Obj Rep 525e1
Coll of Obj Rep 525dn
Coll of Obj Rep 525d1
Coll of Obj Rep 525cn
Coll of Obj Rep 525c1
Coll of Obj Rep 525bn
Coll of Obj Rep 525b1
Coll of Obj Rep 525an
Coll of Obj Rep 525a1
93
Decision-making Unit
540
125
Collection of Sequences
533wa
533wb
533wc
533wd
Sequence
Knowledge Cell
800ca
800cb
800da
800db
800dc
800dd
800de
530c

FIG. 29

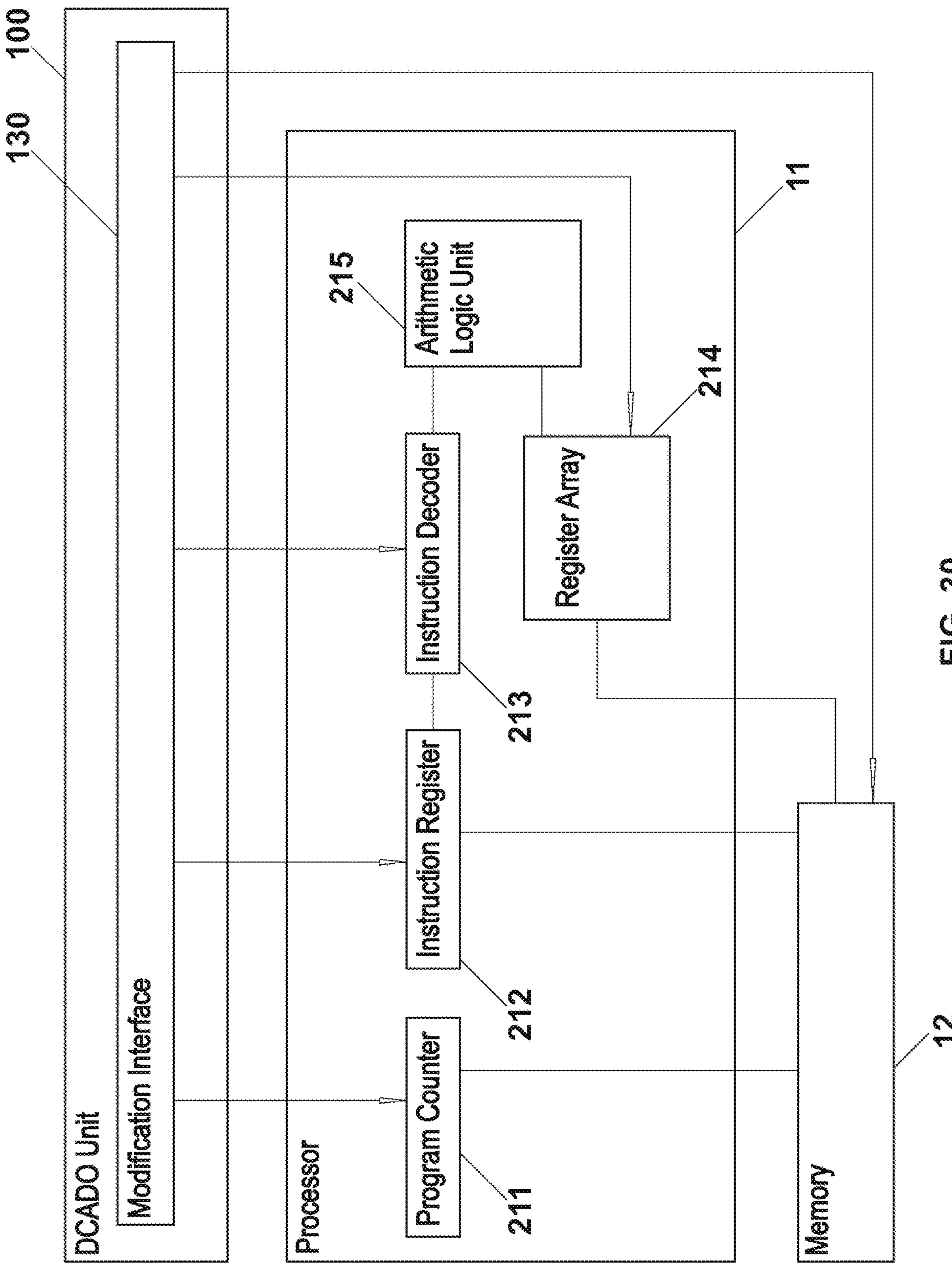

100
130
DCADO Unit
Modification Interface
Processor
Program Counter
211
Instruction Register
212
Instruction Decoder
213
Register Array
214
Arithmetic Logic Unit
215
11
Memory
12

FIG. 30

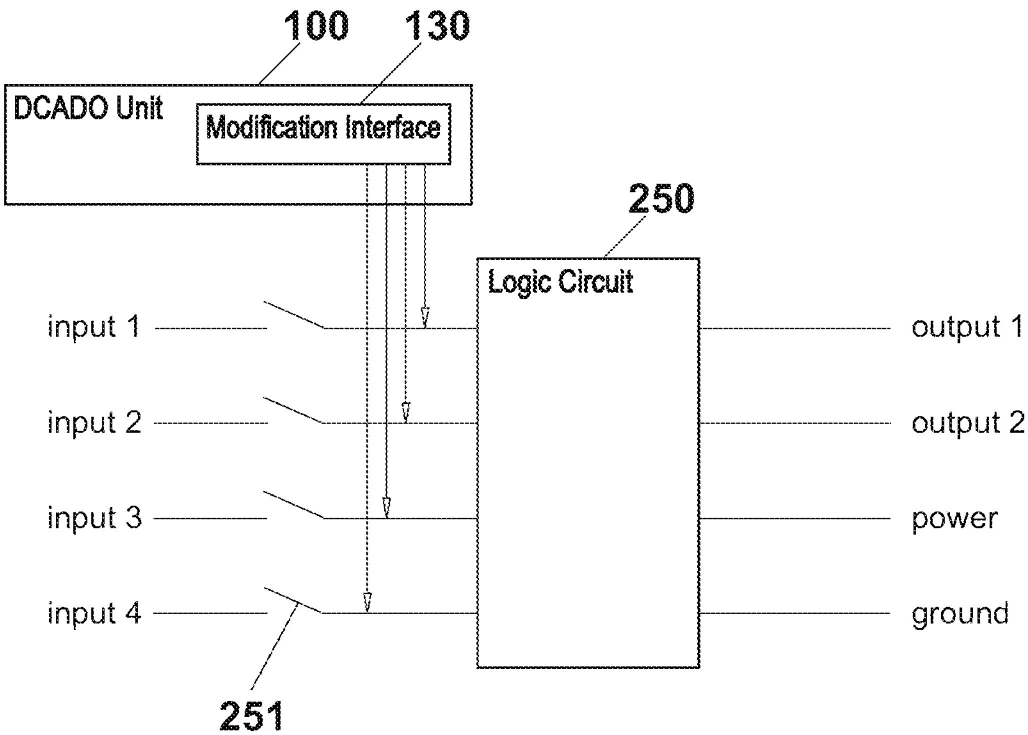


FIG. 31A

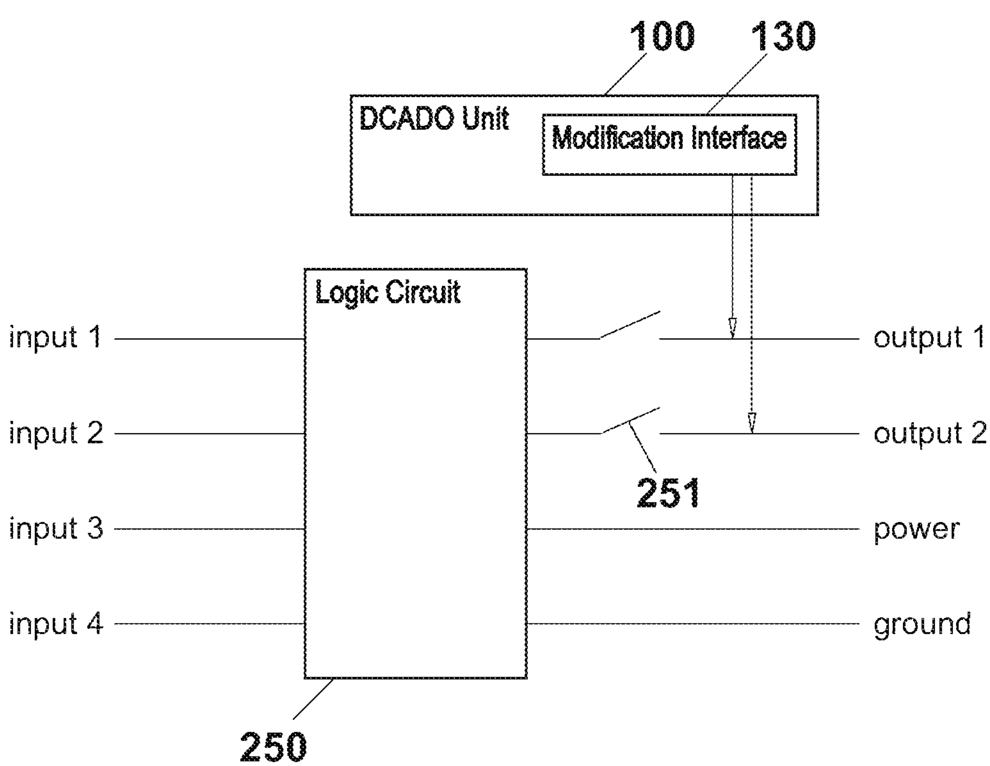
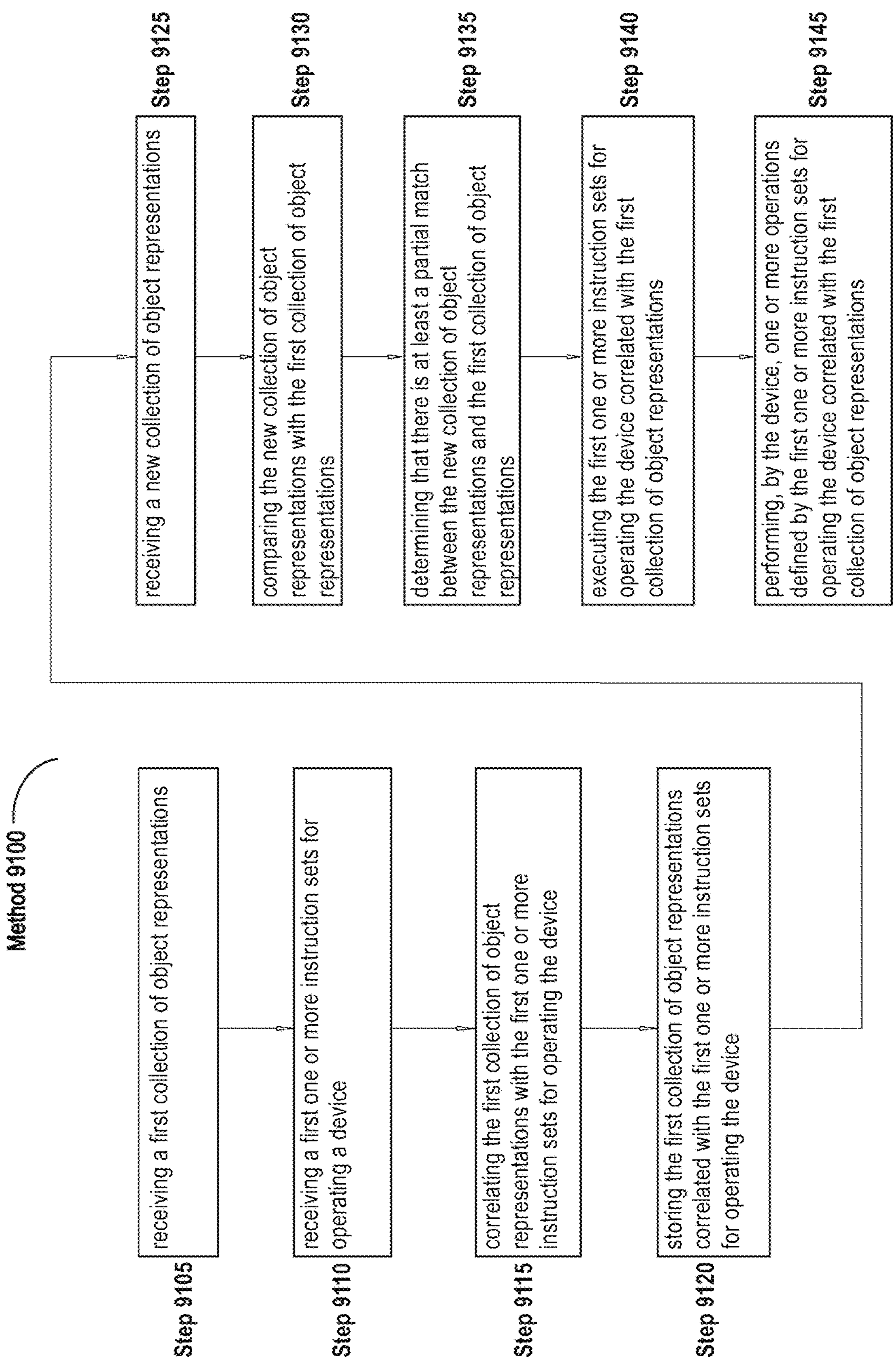


FIG. 31B



Method 9100
receiving a first collection of object representations
Step 9105
receiving a first one or more instruction sets for operating a device
Step 9110
correlating the first collection of object representations with the first one or more instruction sets for operating the device
Step 9115
storing the first collection of object representations correlated with the first one or more instruction sets for operating the device
Step 9120
receiving a new collection of object representations
Step 9125
comparing the new collection of object representations with the first collection of object representations
Step 9130
determining that there is at least a partial match between the new collection of object representations and the first collection of object representations
Step 9135
executing the first one or more instruction sets for operating the device correlated with the first collection of object representations
Step 9140
performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations
Step 9145

FIG. 32

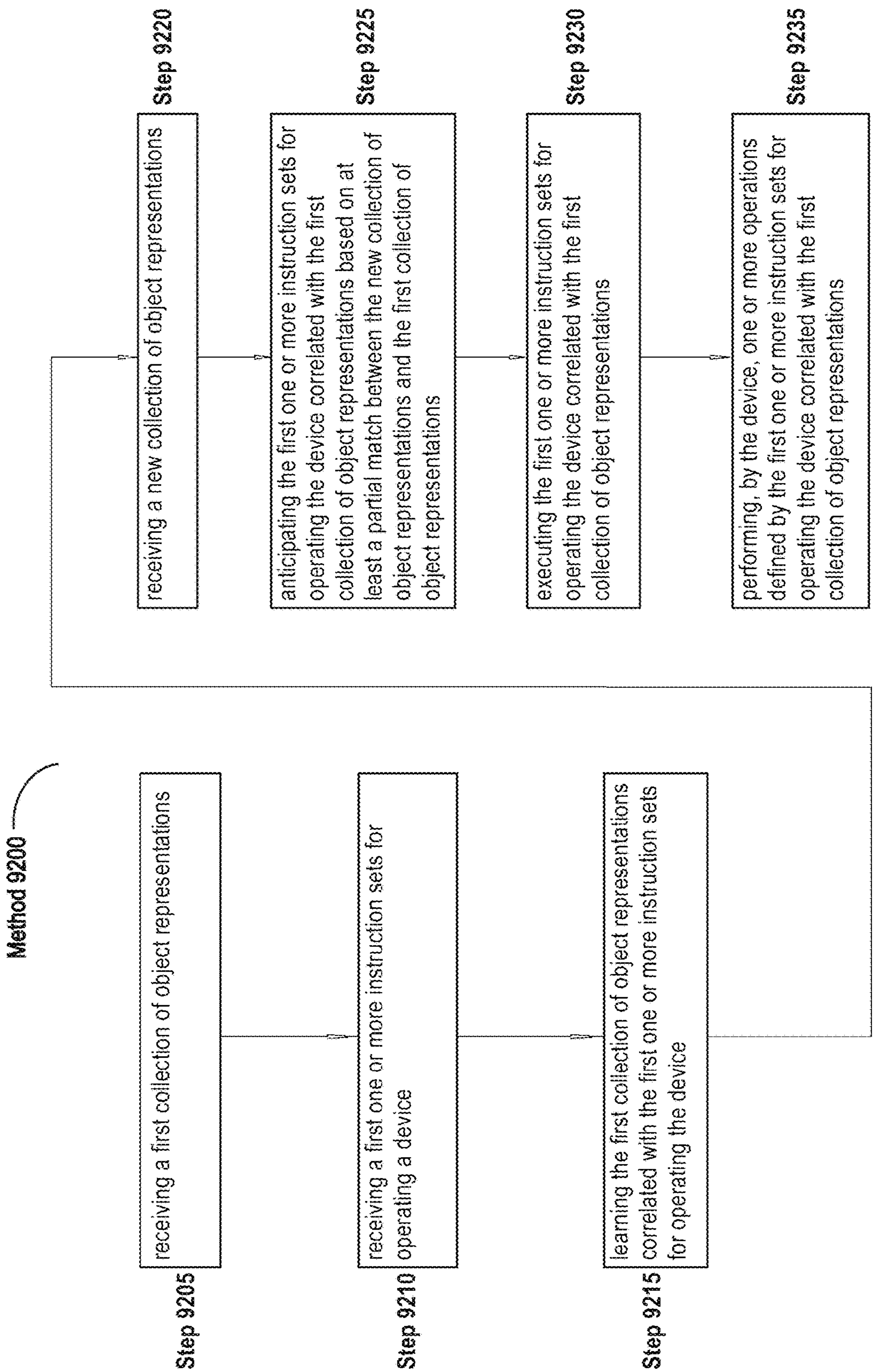
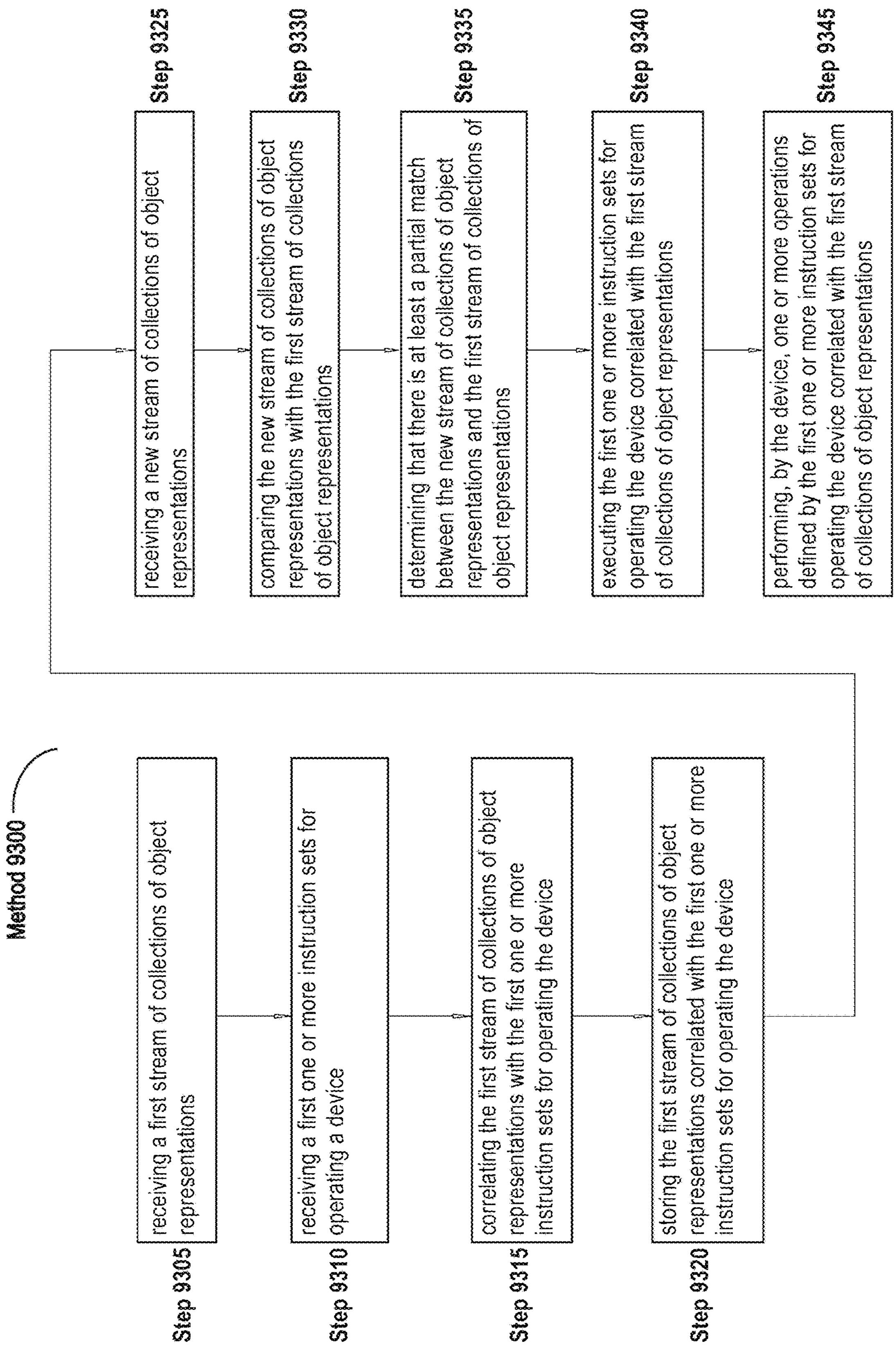
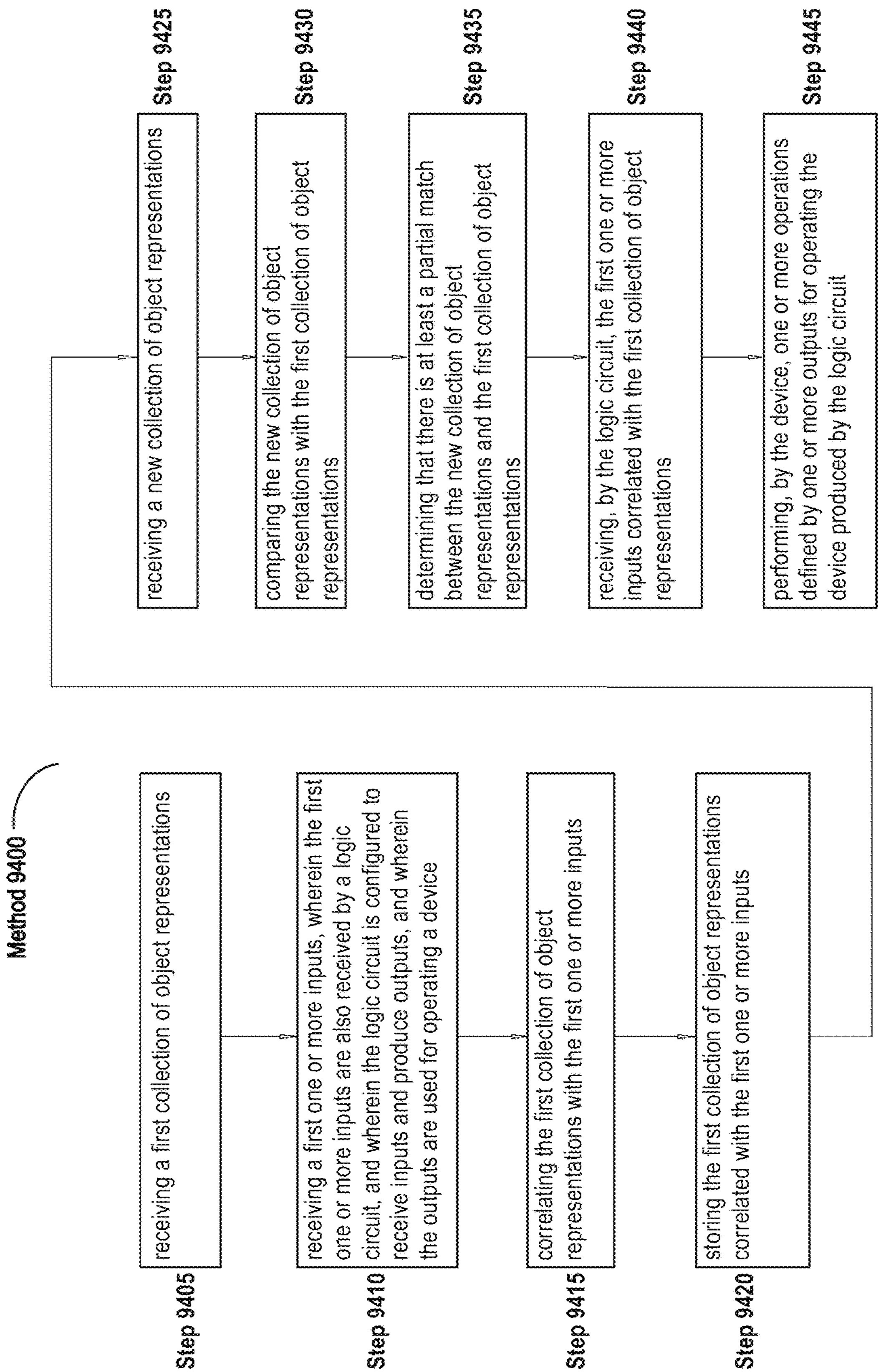

Method 9200
Step 9205
receiving a first collection of object representations
Step 9210
receiving a first one or more instruction sets for operating a device
Step 9215
learning the first collection of object representations correlated with the first one or more instruction sets for operating the device
Step 9220
receiving a new collection of object representations
Step 9225
anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations
Step 9230
executing the first one or more instruction sets for operating the device correlated with the first collection of object representations
Step 9235
performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations

FIG. 33



Method 9300
Step 9305
receiving a first stream of collections of object representations
Step 9310
receiving a first one or more instruction sets for operating a device
Step 9315
correlating the first stream of collections of object representations with the first one or more instruction sets for operating the device
Step 9320
storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device
Step 9325
receiving a new stream of collections of object representations
Step 9330
comparing the new stream of collections of object representations with the first stream of collections of object representations
Step 9335
determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations
Step 9340
executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations
Step 9345
performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations

FIG. 34



Method 9400
Step 9405
receiving a first collection of object representations
Step 9410
receiving a first one or more inputs, wherein the first one or more inputs are also received by a logic circuit, and wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device
Step 9415
correlating the first collection of object representations with the first one or more inputs
Step 9420
storing the first collection of object representations correlated with the first one or more inputs
Step 9425
receiving a new collection of object representations
Step 9430
comparing the new collection of object representations with the first collection of object representations
Step 9435
determining that there is at least a partial match between the new collection of object representations and the first collection of object representations
Step 9440
receiving, by the logic circuit, the first one or more inputs correlated with the first collection of object representations
Step 9445
performing, by the device, one or more operations defined by one or more outputs for operating the device produced by the logic circuit

FIG. 35

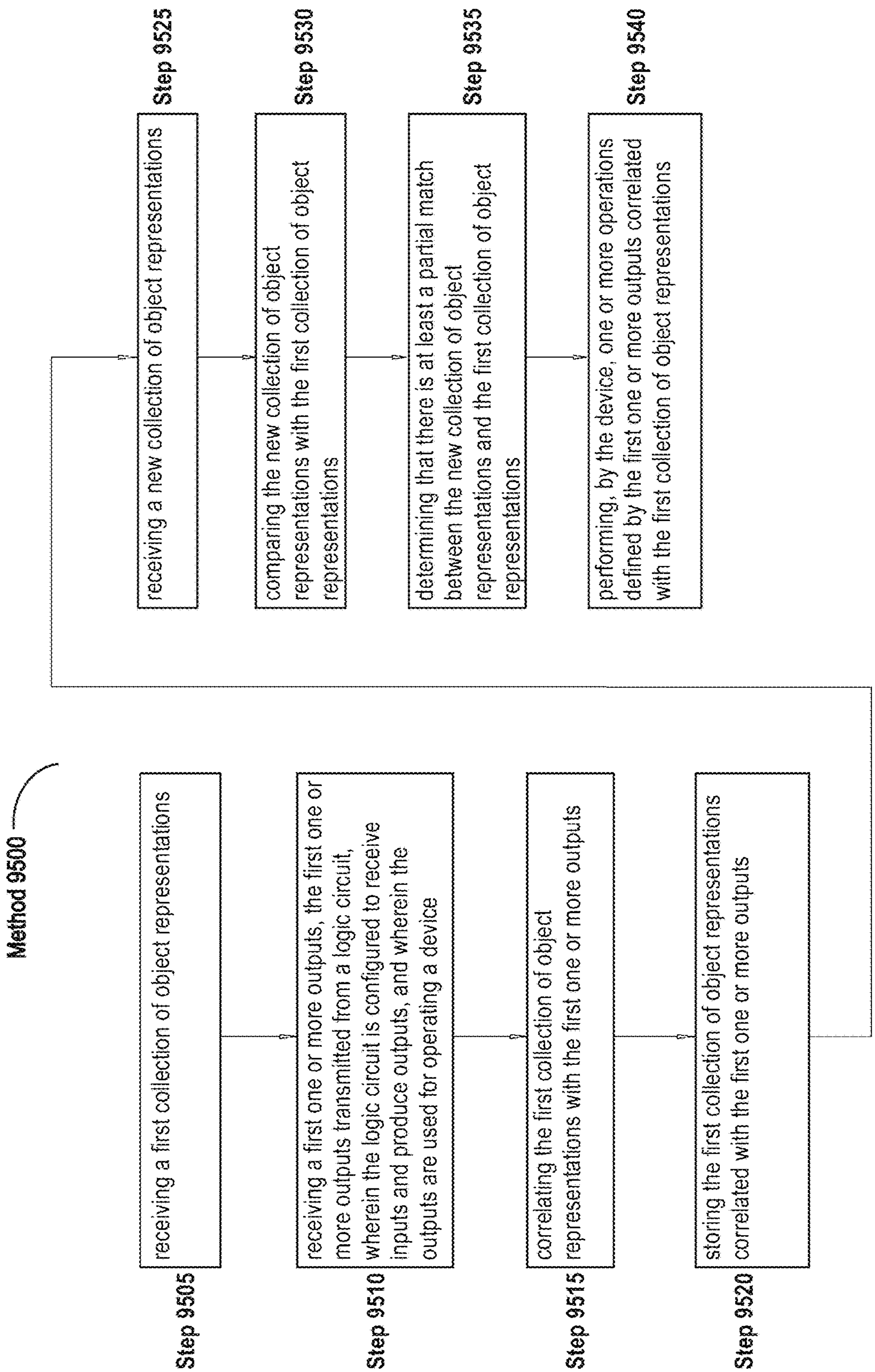

Method 9500
Step 9505
receiving a first collection of object representations
Step 9510
receiving a first one or more outputs, the first one or more outputs transmitted from a logic circuit, wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device
Step 9515
correlating the first collection of object representations with the first one or more outputs
Step 9520
storing the first collection of object representations correlated with the first one or more outputs
Step 9525
receiving a new collection of object representations
Step 9530
comparing the new collection of object representations with the first collection of object representations
Step 9535
determining that there is at least a partial match between the new collection of object representations and the first collection of object representations
Step 9540
performing, by the device, one or more operations defined by the first one or more outputs correlated with the first collection of object representations

FIG. 36

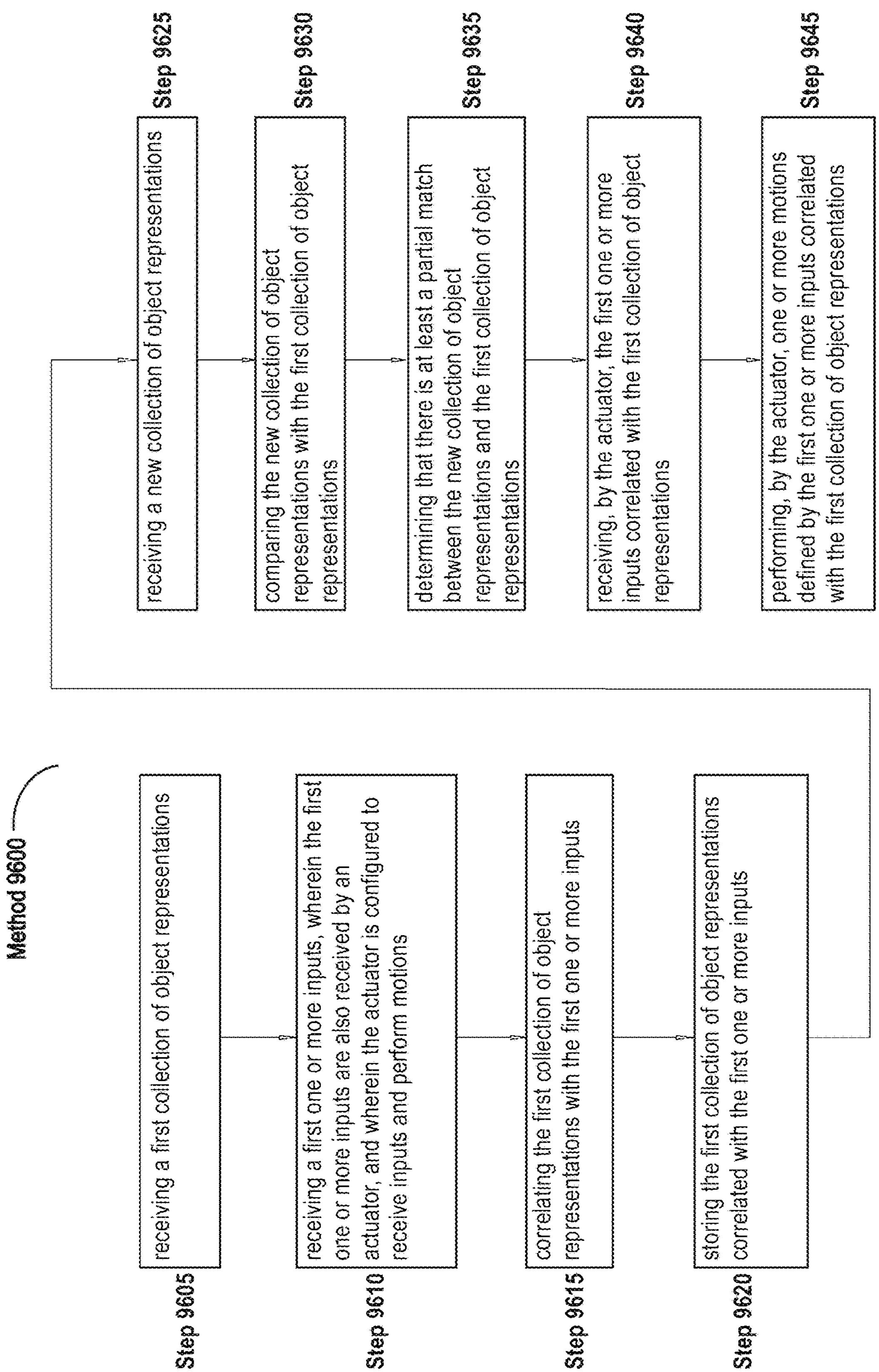

Method 9600
Step 9605
receiving a first collection of object representations
Step 9610
receiving a first one or more inputs, wherein the first one or more inputs are also received by an actuator, and wherein the actuator is configured to receive inputs and perform motions
Step 9615
correlating the first collection of object representations with the first one or more inputs
Step 9620
storing the first collection of object representations correlated with the first one or more inputs
Step 9625
receiving a new collection of object representations
Step 9630
comparing the new collection of object representations with the first collection of object representations
Step 9635
determining that there is at least a partial match between the new collection of object representations and the first collection of object representations
Step 9640
receiving, by the actuator, the first one or more inputs correlated with the first collection of object representations
Step 9645
performing, by the actuator, one or more motions defined by the first one or more inputs correlated with the first collection of object representations

FIG. 37

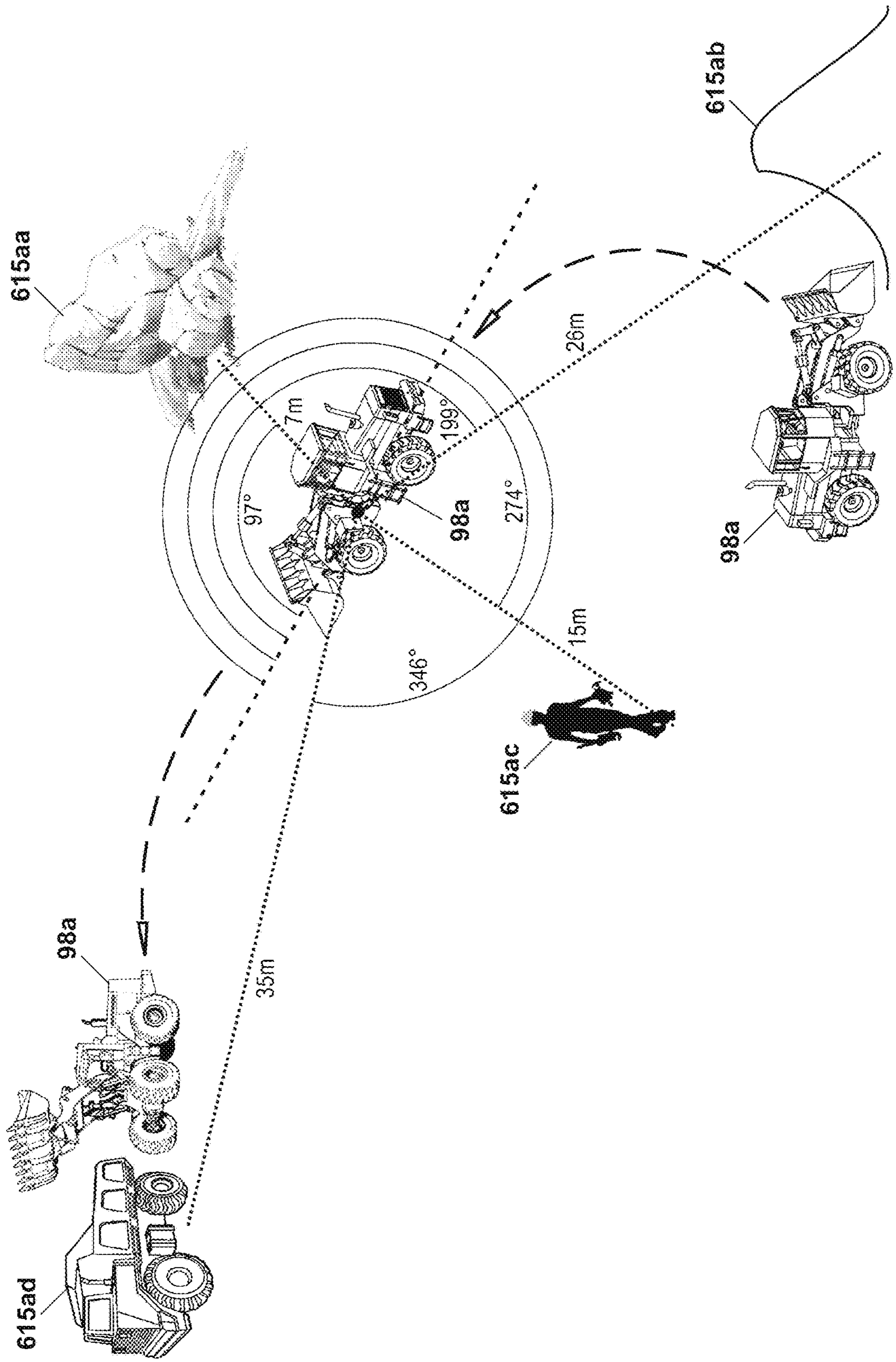

615aa
615ab
26m
7m
199°
97°
98a
274°
15m
346°
98a
615ac
98a
35m
615ad

FIG. 38

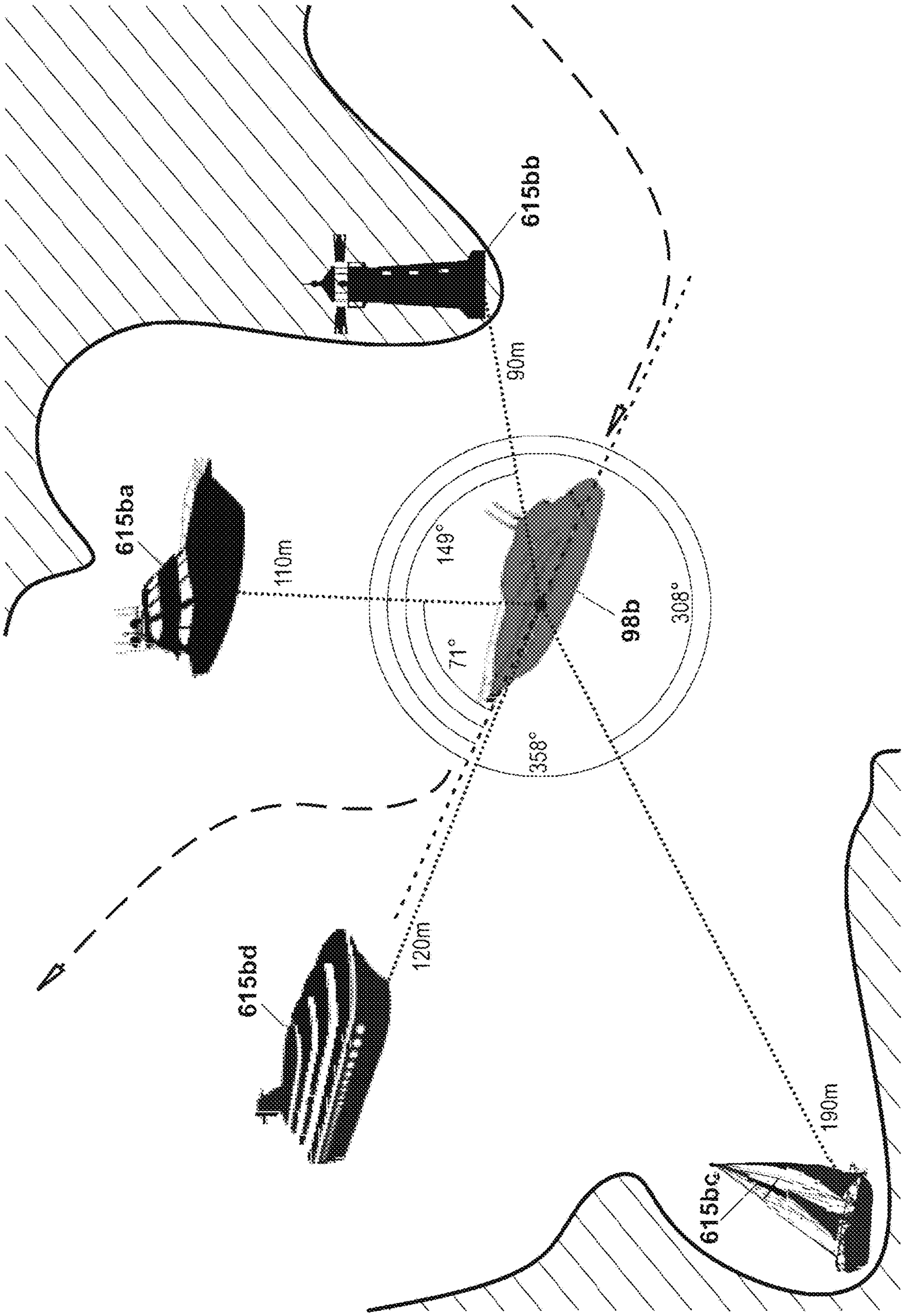


FIG. 39

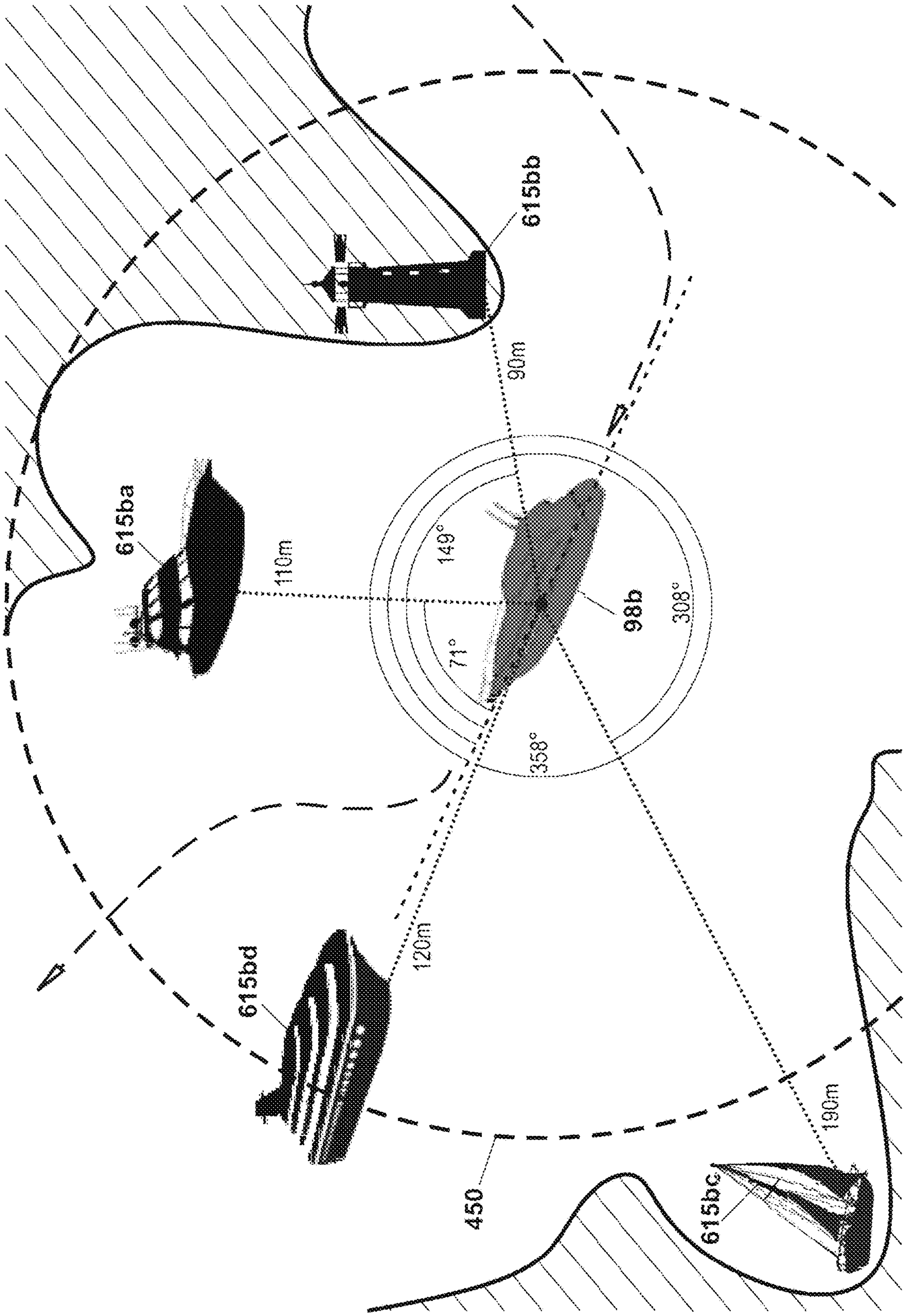

615bb
615ba
90m
110m
149°
71°
98b
308°
358°
120m
615bd
450
190m
615bc

FIG. 40

ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION

CROSS REFERENCE TO RELATED APPLICATION

This application is a continuation of, and claims priority under 35 U.S.C. § 120 from, nonprovisional U.S. patent application Ser. No. 15/340,991 entitled "ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION", filed on Nov. 2, 2016. The disclosure of the foregoing document is incorporated herein by reference.

FIELD

The disclosure generally relates to computing enabled devices and/or systems. The disclosure includes devices, apparatuses, systems, and related methods for providing advanced learning, anticipating, decision making, automation, and/or other functionalities.

COPYRIGHT NOTICE

BACKGROUND

Devices or systems commonly operate by receiving a user's operating directions in various circumstances. Instructions are then executed to effect the operation of a device or system based on user's operating directions. Hence, devices or systems rely on the user to direct their behaviors. Commonly employed device or system operating techniques lack a way to learn operation of a device or system and enable autonomous operation of a device or system.

SUMMARY

In some aspects, the disclosure relates to a system for learning and using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises a processor circuit configured to execute instruction sets for operating a device. The system may further include a memory unit configured to store data. The system may further include a sensor configured to detect objects. The system may further include an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first collection of object representations, the first collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to learn the first collection of object representations

correlated with the first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: receive a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations performed in response to the executing by the processor circuit.

In some embodiments, at least one of: the processor circuit, the memory unit, the sensor, or the artificial intelligence unit are part of, operating on, or coupled to the device. In further embodiments, the device includes one or more devices. In further embodiments, the device includes a construction machine, an assembly machine, an object handling machine, an object dispensing machine, a sorting machine, a restocking machine, an industrial machine, an agricultural machine, a harvesting machine, an appliance, a toy, a robot, a ground vehicle, an aerial vehicle, an aquatic vehicle, a computer, a smartphone, a control device, or a computing enabled device. In further embodiments, the processor circuit includes one or more processor circuits. In further embodiments, the processor circuit includes a logic circuit. The logic circuit may include a microcontroller. The one or more instruction sets may include one or more inputs into or one or more outputs from the logic circuit.

In certain embodiments, the processor circuit includes a logic circuit, the instruction sets for operating the device include inputs into the logic circuit, and executing instruction sets for operating the device includes performing logic operations on the inputs into the logic circuit and producing outputs for operating the device. The logic circuit includes a microcontroller.

In some embodiments, the processor circuit includes a logic circuit, the instruction sets for operating the device include outputs from the logic circuit for operating the device, and executing instruction sets for operating the device includes performing logic operations on inputs into the logic circuit and producing the outputs from the logic circuit for operating the device.

In certain embodiments, the memory unit includes one or more memory units. In further embodiments, the memory unit resides on a remote computing device or a remote computing system, the remote computing device or the remote computing system coupled to the processor circuit via a network or an interface. The remote computing device or the remote computing system may include a server, a cloud, a computing device, or a computing system accessible over the network or the interface.

In some embodiments, the sensor includes one or more sensors. In further embodiments, the sensor includes a camera, a microphone, a lidar, a radar, a sonar, or a detector. In further embodiments, the sensor is part of a remote device. In further embodiments, the sensor is configured to detect objects in the device's surrounding.

In certain embodiments, the artificial intelligence unit is coupled to the sensor. In further embodiments, the artificial intelligence unit is coupled to the memory unit. In further embodiments, the artificial intelligence unit is part of, operating on, or coupled to the processor circuit. In further embodiments, the system further comprises: an additional processor circuit, wherein the artificial intelligence unit is part of, operating on, or coupled to the additional processor circuit. In further embodiments, the artificial intelligence unit is part of, operating on, or coupled to a remote computing device or a remote computing system, the remote computing device or the remote computing system coupled to the processor circuit via a network or an interface. In further embodiments, the artificial intelligence unit includes a circuit, a computing apparatus, or a computing system attachable to the processor circuit. In further embodiments, the artificial intelligence unit includes a circuit, a computing apparatus, or a computing system attachable to the device. In further embodiments, the artificial intelligence unit is attachable to an application for operating the device, the application running on the processor circuit. In further embodiments, the artificial intelligence unit includes a circuit, a computing apparatus, or a computing system built into the processor circuit. In further embodiments, the artificial intelligence unit includes a circuit, a computing apparatus, or a computing system built into the device. In further embodiments, the artificial intelligence unit is built into an application for operating the device, the application running on the processor circuit. In further embodiments, the artificial intelligence unit is provided as a feature of the processor circuit. In further embodiments, the artificial intelligence unit is provided as a feature of an application running on the processor circuit. In further embodiments, the artificial intelligence unit is provided as a feature of the device. In further embodiments, the artificial intelligence unit is further configured to: take control from, share control with, or release control to the processor circuit. In further embodiments, the artificial intelligence unit is further configured to: take control from, share control with, or release control to an application or an object of the application, the application running on the processor circuit.

In some embodiments, the first collection of object representations includes one or more representations of objects detected by the sensor at a time. In further embodiments, the new collection of object representations includes one or more representations of objects detected by the sensor at a time. In further embodiments, the first collection of object representations includes a stream of collections of object representations. In further embodiments, the new collection of object representations includes a stream of collections of object representations. In further embodiments, the first or the new collection of object representations includes or is associated with a time stamp, an order, or a time related information. In further embodiments, the first or the new collection of object representations includes one or more representations of objects in the device's surrounding. In further embodiments, the first or the new collection of object representations includes one or more representations of objects in a remote device's surrounding. In further embodiments, an object representation of the one or more object representations includes one or more object properties. In further embodiments, the first or the new collection of object representations includes one or more object properties. In further embodiments, the first collection of object representations includes a comparative collection of object representations whose at least one portion can be used for comparisons with at least one portion of collections of object representations subsequent to the first collection of object representations, the collections of object representations subsequent to the first collection of object representations comprising the new collection of object representations. In further embodiments, the first collection of object representations includes a comparative collection of object representations that can be used for comparisons with the new collection of object representations. In further embodiments, the new collection of object representations includes an anticipatory collection of object representations whose correlated one or more instruction sets can be used for anticipation of one or more instruction sets to be executed by the processor circuit.

In certain embodiments, the first one or more instruction sets for operating the device include one or more instruction sets that temporally correspond to the first collection of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first collection of object representations include one or more instruction sets executed at a time of generating the first collection of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first collection of object representations include one or more instruction sets executed prior to generating the first collection of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first collection of object representations include one or more instruction sets executed within a threshold period of time prior to generating the first collection of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first collection of object representations include one or more instruction sets executed subsequent to generating the first collection of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first collection of object representations include one or more instruction sets executed within a threshold period of time subsequent to generating the first collection of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first collection of object representations include one or more instruction sets executed within a threshold period of time prior to generating the first collection of object representations or a threshold period of time subsequent to generating the first collection of object representations.

In some embodiments, the first one or more instruction sets for operating the device are executed by the processor circuit. In further embodiments, the first one or more instruction sets for operating the device are part of an application for operating the device, the application running on the processor circuit. In further embodiments, the first one or more instruction sets for operating the device include one or more inputs into or one or more outputs from the processor circuit. In further embodiments, the first one or more instruction sets for operating the device include values or states of one or more registers or elements of the processor circuit. In further embodiments, the first one or more instruction sets for operating the device include at least one of: a command, a keyword, a symbol, an instruction, an operator, a variable, a value, an object, a data structure, a function, a parameter, a state, a signal, an input, an output, a character, a digit, or a reference thereto. In further embodiments, the first one or more instruction sets for operating the device include a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, a structured query language (SQL) code, or a machine code. In further embodiments, the first one or

more instruction sets for operating the device include one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the processor circuit includes a logic circuit. The first one or more instruction sets for operating the device may include one or more inputs into a logic circuit. The first one or more instruction sets for operating the device may include one or more outputs from a logic circuit.

In certain embodiments, the first one or more instruction sets for operating the device include one or more instruction sets for operating an application or an object of the application, the application running on the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes obtaining the first one or more instruction sets for operating the device from the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device as they are executed by the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from a register or an element of the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from an element that is part of, operating on, or coupled to the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from at least one of: the memory unit, the device, a virtual machine, a runtime engine, a hard drive, a storage device, a peripheral device, a network connected device, or a user. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from a plurality of processor circuits, applications, memory units, devices, virtual machines, runtime engines, hard drives, storage devices, peripheral devices, network connected devices, or users.

In some embodiments, the processor circuit includes a logic circuit, and wherein the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from the logic circuit. In further embodiments, the logic circuit includes a microcontroller. In further embodiments, the receiving the first one or more instruction sets for operating the device from the logic circuit includes receiving the first one or more instruction sets for operating the device from an element of the logic circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device from the logic circuit includes receiving one or more inputs into the logic circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device from the logic circuit includes receiving one or more outputs from the logic circuit.

In certain embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from an application for operating the device, the application running on the processor circuit.

In some embodiments, the system further comprises: an application including instruction sets for operating the device, the application running on the processor circuit, wherein the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from the application.

In certain embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, a structured query language (SQL) code, or a machine code. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of an element that is part of, operating on, or coupled to the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a register of the processor circuit, the memory unit, a storage, or a repository where the first one or more instruction sets for operating the device are stored. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of the processor circuit, the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a processing element. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of the processor circuit or tracing, profiling, or instrumentation of a component of the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of an application or an object of the application, the application running on the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of one or more of code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a user input. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: a manual, an automatic, a dynamic, or a just in time (JIT) tracing, profiling, or instrumentation. In further embodiments, the receiving the first one or more instruction sets for operating the device includes utilizing at least one of: a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, a logging tool, or an independent tool for obtaining instruction sets. In further embodiments, the receiving the first one or more instruction sets for operating the device includes utilizing an assembly language. In further embodiments, the receiving the first one or more instruction sets for operating the device includes utilizing a branch or a jump. In further embodiments, the receiving the first one or more instruction sets for operating the device includes a branch tracing or a simulation tracing. In further

embodiments, the system further comprises: an interface configured to receive instruction sets, wherein the first one or more instruction sets for operating the device are received by the interface. The interface may include an acquisition interface.

In some embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the device include a knowledge of how the device operated in a circumstance. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the device are included in a neuron, a node, a vertex, or an element of a knowledgebase. In further embodiments, the knowledgebase includes a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. In further embodiments, some of the neurons, nodes, vertices, or elements are interconnected. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the device are structured into a knowledge cell. In further embodiments, the knowledge cell is included in a neuron, a node, a vertex, or an element of a knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device includes correlating the first collection of object representations with the first one or more instruction sets for operating the device. In further embodiments, the correlating the first collection of object representations with the first one or more instruction sets for operating the device includes generating a knowledge cell, the knowledge cell comprising the first collection of object representations correlated with the first one or more instruction sets for operating the device. In further embodiments, the correlating the first collection of object representations with the first one or more instruction sets for operating the device includes structuring a knowledge of how the device operated in a circumstance. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device includes learning a user's knowledge, style, or methodology of operating the device in a circumstance.

In certain embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device includes storing the first collection of object representations correlated with the first one or more instruction sets for operating the device into the memory unit, the first collection of object representations correlated with the first one or more instruction sets for operating the device being part of a plurality of collections of object representations correlated with one or more instruction sets for operating the device stored in the memory unit. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the device include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the device are organized into a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, each of the plurality of collections of object representations correlated with one or more instruction sets for operating the device is included in a neuron, a node, a vertex, or an element of a knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the device include a user's knowledge, style, or methodology of operating the device in circumstances. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the device are stored on a remote computing device or a remote computing system. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the device include an artificial intelligence system for knowledge structuring, storing, or representation. The artificial intelligence system for knowledge structuring, storing, or representation may include at least one of: a deep learning system, a supervised learning system, an unsupervised learning system, a neural network, a search-based system, an optimization-based system, a logic-based system, a fuzzy logic-based system, a tree-based system, a graph-based system, a hierarchical system, a symbolic system, a sub-symbolic system, an evolutionary system, a genetic system, a multi-agent system, a deterministic system, a probabilistic system, or a statistical system.

In some embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes comparing at least one portion of the new collection of object representations with at least one portion of the first collection of object representations. In further embodiments, the at least one portion of the new collection of object representations include at least one object representation or at least one object property of the new collection of object representations. In further embodiments, the at least one portion of the first collection of object representations include at least one object representation or at least one object property of the first collection of object representations. In further embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes comparing at least one object representation from the new collection of object representations with at least one object representation from the first collection of object representations. In further embodiments, the comparing at least one object representation from the new collection of object representations with at least one object representation from the first collection of object representations includes comparing at least one object property of the at least one object representation from the new collection of object representations with at least one object property of the at least one object representation from the first collection of object representations.

In certain embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object repre-

sentations and the first collection of object representations includes determining that there is at least a partial match between the new collection of object representations and the first collection of object representations. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that there is at least a partial match between one or more portions of the new collection of object representations and one or more portions of the first collection of object representations. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that a similarity between at least one portion of the new collection of object representations and at least one portion of the first collection of object representations exceeds a similarity threshold. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining a substantial similarity between at least one portion of the new collection of object representations and at least one portion of the first collection of object representations. The substantial similarity may be achieved when a similarity between the at least one portion of the new collection of object representations and the at least one portion of the first collection of object representations exceeds a similarity threshold. The substantial similarity may be achieved when a number or a percentage of matching or partially matching portions of the new collection of object representations and portions of the first collection of object representations exceeds a threshold number or threshold percentage. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that a number or a percentage of matching or partially matching object representations from the new collection of object representations and from the first collection of object representations exceeds a threshold number or threshold percentage. The matching or partially matching object representations from the new collection of object representations and from the first collection of object representations may be determined factoring in at least one of: a type of an object representation, an importance of an object representation, a threshold for a similarity in an object representation, or a threshold for a difference in an object representation. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that a number or a percentage of matching or partially matching object properties from the new collection of object representations and from the first collection of object representations exceeds a threshold number or threshold percentage. The matching or partially matching object properties from the new collection of object representations and from the first collection of object representations may be determined factoring in at least one of: an association of an object property with an object representation, a category of an object property, an importance of an object property, a threshold for a similarity in an object property, or a threshold for a difference in an object property. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that there is at least a partial match between at least one object representation from the new collection of object representations and at least one object representation from the first collection of object representations. The determining that there is at least a partial match between at least one object representation from the new collection of object representations and at least one object representation from the first collection of object representations includes determining that there is at least a partial match between at least one object property of the at least one object representation from the new collection of object representations and at least one object property of the at least one object representation from the first collection of object representations.

In certain embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations instead of or prior to an instruction set that would have been executed next. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first collection of object representations into a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting the processor circuit to the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting the processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes transmitting, to the processor circuit for execution, the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes issuing an interrupt to the processor circuit and executing the first one or more instruction sets for operating the device correlated with the first collection of object representations following the interrupt. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying an element that is part of, operating on, or coupled to the processor circuit.

In some embodiments, the processor circuit includes a logic circuit, and wherein the causing the processor circuit

to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the logic circuit includes a microcontroller. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying an element of the logic circuit. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first collection of object representations into an element of the logic circuit. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting the logic circuit to the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes replacing inputs into the logic circuit with the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes replacing outputs from the logic circuit with the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In certain embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes causing an application for operating the device to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations, the application running on the processor circuit.

In some embodiments, the system further comprises: an application including instruction sets for operating the device, the application running on the processor circuit, wherein the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying the application.

In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting an application to the first one or more instruction sets for operating the device correlated with the first collection of object representations, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting an application to one or more alternate instruction sets, the application running on the processor circuit, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying at least one of: the memory unit, a register of the processor circuit, a storage, or a repository where instruction sets are stored or used. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets for operating an application or an object of the application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying at least one of: an element of the processor circuit, an element of the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing an assembly language. In further embodiments, the causing the processor circuit to

execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes adding or inserting additional code into a code of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes at least one of: modifying, removing, rewriting, or overwriting a code of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of an application, the application running on the processor circuit. The branching or redirecting the code may include inserting at least one of: a branch, a jump, or a means for redirecting an execution. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes implementing a user's knowledge, style, or methodology of operating the device in a circumstance.

In certain embodiments, the system further comprises: an interface configured to cause execution of instruction sets, wherein the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations is caused by the interface. The interface may include a modification interface.

In some embodiments, the one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations include one or more operations with or by a computing enabled device. In further embodiments, the performing the one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations includes implementing a user's knowledge, style, or methodology of operating the device in a circumstance.

In certain embodiments, the system further comprising: an application running on the processor circuit.

In some embodiments, the instruction sets for operating the device are part of an application for operating the device, the application running on the processor circuit.

In certain embodiments, the system of further comprises: an application for operating the device, the application running on the processor circuit. The application for operating the device may include the instruction sets for operating the device.

In some embodiments, the artificial intelligence unit is further configured to: receive at least one extra information. In further embodiments, the at least one extra information include one or more of: a time information, a location information, a computed information, or a contextual information. In further embodiments, the at least one extra information include one or more of: an information on an object, an information on an object representation, an information on a collection of object representations, an information on a device's circumstance, an information on an instruction set, an information on an application, an information on the processor circuit, an information on the device, or an information on an user. In further embodiments, the artificial intelligence unit is further configured to: learn the first collection of object representations correlated with the at least one extra information. The learning the first collection of object representations correlated with at least one extra information may include correlating the first collection of object representations with the at least one extra information. The learning the first collection of object representations correlated with at least one extra information may include storing the first collection of object representations correlated with the at least one extra information into the memory unit. In further embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations. The anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations may include comparing an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations. The anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations may include determining that a similarity between an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations exceeds a similarity threshold.

In certain embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: cause the user interface to present a user with an option to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In some embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: receive, via the user interface, a user's selection to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In certain embodiments, the artificial intelligence unit is further configured to: rate the executed first one or more instruction sets for operating the device correlated with the first collection of object representations. The rating the executed first one or more instruction sets for operating the device correlated with the first collection of object representations may include causing a user interface to display the executed first one or more instruction sets for operating the device correlated with the first collection of object representations along with one or more rating values as options to be selected by a user. The rating the executed first one or more instruction sets for operating the device correlated with the first collection of object representations may include rating the executed first one or more instruction sets for operating the device correlated with the first collection of object representations without a user input.

In some embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: cause the user interface to present a user with an option to cancel the execution of the executed first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the canceling the execution of the executed first one or more instruction sets for operating the device correlated with the first collection of object representations includes restoring the processor circuit or the device to a prior state. The restoring the processor circuit or the device to a prior state may include saving the state of the processor circuit or the device prior to executing the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In certain embodiments, the system further comprises: an input device configured to receive a user's operating directions, the user's operating directions for instructing the processor circuit on how to operate the device.

In some embodiments, the autonomous device operating includes a partially or a fully autonomous device operating. The partially autonomous device operating may include executing the first one or more instruction sets for operating the device correlated with the first collection of object representations responsive to a user confirmation. In further embodiments, the fully autonomous device operating may include executing the first one or more instruction sets for operating the device correlated with the first collection of object representations without a user confirmation.

In certain embodiments, the artificial intelligence unit is further configured to: receive a second collection of object representations, the second collection of object representations including one or more representations of objects detected by the sensor; receive a second one or more instruction sets for operating the device; and learn the second collection of object representations correlated with the second one or more instruction sets for operating the device. In further embodiments, the second collection of object representations includes one or more representations of objects detected by the sensor at a time. In further embodiments, the second collection of object representations includes a stream of collections of object representations. In further embodiments, the second collection of object representations includes or is associated with a time stamp, an order, or a time related information. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device and the learning the second collection of object representations correlated with the second one or more instruction sets for operating the device include creating a connection between the first collection of

object representations correlated with the first one or more instruction sets for operating the device and the second collection of object representations correlated with the second one or more instruction sets for operating the device. The connection may include or be associated with at least one of: an occurrence count, a weight, a parameter, or a data. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device and the learning the second collection of object representations correlated with the second one or more instruction sets for operating the device include updating a connection between the first collection of object representations correlated with the first one or more instruction sets for operating the device and the second collection of object representations correlated with the second one or more instruction sets for operating the device. The updating the connection between the first collection of object representations correlated with the first one or more instruction sets for operating the device and the second collection of object representations correlated with the second one or more instruction sets for operating the device may include updating at least one of: an occurrence count, a weight, a parameter, or a data included in or associated with the connection. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device includes storing the first collection of object representations correlated with the first one or more instruction sets for operating the device into a first node of a knowledgebase, and wherein the learning the second collection of object representations correlated with the second one or more instruction sets for operating the device includes storing the second collection of object representations correlated with the second one or more instruction sets for operating the device into a second node of the knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. The knowledgebase may be stored in the memory unit. The learning the first collection of object representations correlated with the first one or more instruction sets for operating the device and the learning the second collection of object representations correlated with the second one or more instruction sets for operating the device include creating a connection between the first node and the second node. The learning the first collection of object representations correlated with the first one or more instruction sets for operating the device and the learning the second collection of object representations correlated with the second one or more instruction sets for operating the device include updating a connection between the first node and the second node. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the device is stored into a first node of a neural network and the second collection of object representations correlated with the second one or more instruction sets for operating the device is stored into a second node of the neural network. The first node and the second node may be connected by a connection. The first node may be part of a first layer of the neural network and the second node may be part of a second layer of the neural network. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the device is stored into a first node of a graph and the second collection of object representations correlated with the second one or more instruction sets for operating the device is stored into a second node of

the graph. The first node and the second node may be connected by a connection. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the device is stored into a first node of a sequence and the second collection of object representations correlated with the second one or more instruction sets for operating the device is stored into a second node of the sequence.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first collection of object representations, the first collection of object representations including one or more representations of objects detected by a sensor. The operations may further include: receiving a first one or more instruction sets for operating a device. The operations may further include: learning the first collection of object representations correlated with the first one or more instruction sets for operating the device. The operations may further include: receiving a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The operations may further include: anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further include: causing an execution of the first one or more instruction sets for operating the device correlated with the first collection of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations by the device performed in response to the execution.

In certain embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from the one or more processor circuits or from another one or more processor circuits. In further embodiments, the execution of the first one or more instruction sets for operating the device correlated with the first collection of object representations is performed by the one or more processor circuits or by another one or more processor circuits.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a processor circuit, the first collection of object representations including one or more representations of objects detected by a sensor. The method may further include: (b) receiving a first one or more instruction sets for operating a device by the processor circuit. The method may further include: (c) learning the first collection of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the processor circuit. The method may further include: (d) receiving a new collection of object representations by the processor circuit, the new collection of object representations including one or more representations of objects detected by the sensor. The method may further include: (e) anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (e) performed by the processor circuit. The method may further include: (f) executing the first one or more instruction sets for operating the device correlated with the first collection of object representations, the executing of (f) performed in response to the anticipating of (e). The method may further include: (g) performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations by the device performed in response to the executing of (f).

In certain embodiments, the receiving of (b) includes receiving the first one or more instruction sets for operating the device from the processor circuit or from another processor circuit. In further embodiments, the executing of (f) is performed by the processor circuit or by another processor circuit.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable as well as the following embodiments.

In some embodiments, the first one or more instruction sets for operating the device are executed by a processor circuit. In further embodiments, the first one or more instruction sets for operating the device are part of an application for operating the device. In further embodiments, the first one or more instruction sets for operating the device include one or more inputs into or one or more outputs from a processor circuit. In further embodiments, the first one or more instruction sets for operating the device include values or states of one or more registers or elements of a processor circuit. In further embodiments, the first one or more instruction sets for operating the device include one or more inputs into a logic circuit. In further embodiments, the first one or more instruction sets for operating the device include one or more outputs from a logic circuit. In further embodiments, the first one or more instruction sets for operating the device include one or more instruction sets for operating an application or an object of the application.

In certain embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device as they are executed by a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from a register or an element of a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from an element that is part of, operating on, or coupled to a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from at least one of: the memory unit, the device, a virtual machine, a

runtime engine, a hard drive, a storage device, a peripheral device, a network connected device, or a user. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from a plurality of processor circuits, applications, memory units, devices, virtual machines, runtime engines, hard drives, storage devices, peripheral devices, network connected devices, or users.

In some embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from a logic circuit. In further embodiments, the logic circuit includes a microcontroller. In further embodiments, the receiving the first one or more instruction sets for operating the device from the logic circuit includes receiving the first one or more instruction sets for operating the device from an element of the logic circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device from the logic circuit includes receiving one or more inputs into the logic circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device from the logic circuit includes receiving one or more outputs from the logic circuit.

In certain embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from an application for operating the device. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from an application, the application including instruction sets for operating the device. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of an element that is part of, operating on, or coupled to a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a register of a processor circuit, the memory unit, a storage, or a repository where the first one or more instruction sets for operating the device are stored. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a processor circuit, the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a processing element. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a processor circuit or tracing, profiling, or instrumentation of a component of a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of an application or an object of the application. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device by an interface. The interface may include an acquisition interface.

In some embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device includes storing the first collection of object representations correlated with the first one or more instruction sets for operating the device into a memory unit, the first collection of object representations correlated with the first one or more instruction sets for operating the device being part of a plurality of collections of object representations correlated with one or more instruction sets for operating the device stored in the memory unit.

In certain embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes executing the first one or more instruction sets for operating the device correlated with the first collection of object representations instead of or prior to an instruction set that would have been executed next. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying a register or an element of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first collection of object representations into a register or an element of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting a processor circuit to the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting a processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes transmitting, to a processor circuit for execution, the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes issuing an interrupt to a processor circuit and executing the first one or more instruction sets for operating the device correlated with the first collection of object representations following the interrupt. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying an element that is part of, operating on, or coupled to a processor circuit.

In some embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes executing, by a logic circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the logic circuit includes a microcontroller. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying an element of the logic circuit. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations includes inserting the

first one or more instruction sets for operating the device correlated with the first collection of object representations into an element of the logic circuit. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting the logic circuit to the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations includes replacing inputs into the logic circuit with the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations includes replacing outputs from the logic circuit with the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In certain embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes executing, by an application for operating the device, the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying an application, the application including instruction sets for operating the device. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting an application to the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting an application to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying at least one of: the memory unit, a register of a processor circuit, a storage, or a repository where instruction sets are stored or used. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets for operating an application or an object of the application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying at least one of: an element of a processor circuit, an element of the

device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing an assembly language. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes adding or inserting additional code into a code of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes at least one of: modifying, removing, rewriting, or overwriting a code of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of an application. The branching or redirecting the code may include inserting at least one of: a branch, a jump, or a means for redirecting an execution. In further embodiments, the executing the first one or more

instruction sets for operating the device correlated with the first collection of object representations includes implementing a user's knowledge, style, or methodology of operating the device in a circumstance. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes executing the first one or more instruction sets for operating the device correlated with the first collection of object representations via an interface. The interface may include a modification interface.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving at least one extra information. In further embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: learning the first collection of object representations correlated with the at least one extra information.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: presenting, via a user interface, a user with an option to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving, via a user interface, a user's selection to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: rating the executed first one or more instruction sets for operating the device correlated with the first collection of object representations.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: presenting, via a user interface, a user with an option to cancel the execution of the executed first one or more instruction sets for operating the device correlated with the first collection of object representations.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving, via an input device, a user's operating directions, the user's operating directions for instructing a processor circuit on how to operate the device.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving a second collection of object representations, the second collection of object representations including one or more representations of objects detected by the sensor; receiving a second one or more instruction sets for operating the device; and learning the second collection of object representations correlated with the second one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving, by a first processor circuit of the one or more processor circuits, a first collection of object representations, the first collection of object representations including one or more representations of objects detected by a sensor. The operations may further comprise: receiving, by the first processor circuit of the one or more processor circuits, a first one or more instruction sets for operating a device. The operations may further comprise: learning, by the first

processor circuit of the one or more processor circuits, the first collection of object representations correlated with the first one or more instruction sets for operating the device. The operations may further comprise: receiving, by the first processor circuit of the one or more processor circuits, a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The operations may further comprise: anticipating, by the first processor circuit of the one or more processor circuits, the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further comprise: causing, by the first processor circuit of the one or more processor circuits, an execution, by a second processor circuit of the one or more processor circuits, of the first one or more instruction sets for operating the device correlated with the first collection of object representations, the causing performed in response to the anticipating, by the first processor circuit of the one or more processor circuits, the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations by the device performed in response to the execution.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a first processor circuit, the first collection of object representations including one or more representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more instruction sets for operating a device by the first processor circuit. The method may further comprise: (c) learning the first collection of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the first processor circuit. The method may further comprise: (d) receiving a new collection of object representations by the first processor circuit, the new collection of object representations including one or more representations of objects detected by the sensor. The method may further comprise: (e) anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (e) performed by the first processor circuit. The method may further comprise: (f) executing, by a second processor circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations, the executing of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations by the device performed in response to the executing of (f).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps,

and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for learning a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first collection of object representations, the first collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: learn the first collection of object representations correlated with the first one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first collection of object representations, the first collection of object representations including one or more representations of objects detected by a sensor. The operations may further comprise: receiving a first one or more instruction sets for operating a device. The operations may further comprise: learning the first collection of object representations correlated with the first one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a processor circuit, the first collection of object representations including one or more representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more instruction sets for operating a device by the processor circuit. The method may further comprise: (c) learning the first collection of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the processor circuit.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: access the memory unit that comprises a plurality of collections of object representations correlated with one or more instruction sets for operating the device, the plurality of collections of object representations correlated with one or more instruction sets for operating the device including a first collection of object representations correlated with a first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: receive a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations performed in response to the executing by the processor circuit.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: accessing a memory unit that comprises a plurality of collections of object representations correlated with one or more instruction sets for operating a device, the plurality of collections of object representations correlated with one or more instruction sets for operating the device including a first collection of object representations correlated with a first one or more instruction sets for operating the device. The operations may further comprise: receiving a new collection of object representations, the new collection of object representations including one or more representations of objects detected by a sensor. The operations may further comprise: anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further comprise: causing an execution of the first one or more instruction sets for operating the device correlated with the first collection of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations by the device performed in response to the execution.

In some aspects, the disclosure relates to a method comprising: (a) accessing a memory unit that comprises a plurality of collections of object representations correlated with one or more instruction sets for operating a device, the plurality of collections of object representations correlated with one or more instruction sets for operating the device including a first collection of object representations correlated with a first one or more instruction sets for operating the device, the accessing of (a) performed by a processor

circuit. The method may further comprise: (b) receiving a new collection of object representations by the processor circuit, the new collection of object representations including one or more representations of objects detected by a sensor. The method may further comprise: (c) anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (c) performed by the processor circuit. The method may further comprise: (d) executing the first one or more instruction sets for operating the device correlated with the first collection of object representations, the executing of (d) performed in response to the anticipating of (c). The method may further comprise: (e) performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations by the device performed in response to the executing of (d).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for learning and using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first stream of collections of object representations, the first stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: learn the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: receive a new stream of collections of object representations, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations performed in response to the executing by the processor circuit.

In certain embodiments, each collection of object representations includes one or more representations of objects detected by the sensor at a time. In further embodiments, each collection of object representations includes one or more of object representations. In further embodiments, each collection of object representations includes or is associated with a time stamp, an order, or a time related information. In further embodiments, the first stream of collections of object representations includes one or more collections of representations of objects detected by the sensor over time. In further embodiments, the new stream of collections of object representations includes one or more collections of representations of objects detected by the sensor over time. In further embodiments, the first or the new stream of collections of object representations includes one or more collections of representations of objects in the device's surrounding. In further embodiments, the first or the new stream of collections of object representations includes one or more collections of representations of objects in a remote device's surrounding. In further embodiments, an object representation of a stream of collections of object representations includes one or more object properties. In further embodiments, the first or the new stream of collections of object representations includes one or more object properties. In further embodiments, the first stream of collections of object representations includes a comparative stream of collections of object representations whose at least one portion can be used for comparisons with at least one portion of streams of collections of object representations subsequent to the first stream of collections of object representations, the streams of collections of object representations subsequent to the first stream of collections of object representations comprising the new stream of collections of object representations. In further embodiments, the first stream of collections of object representations includes a comparative stream of collections of object representations that can be used for comparisons with the new stream of collections of object representations. In further embodiments, the new stream of collections of object representations includes an anticipatory stream of collections of object representations whose correlated one or more instruction sets can be used for anticipation of one or more instruction sets to be executed by the processor circuit.

In some embodiments, the first one or more instruction sets for operating the device include one or more instruction sets that temporally correspond to the first stream of collections of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first stream of collections of object representations include one or more instruction sets executed at a time of generating the first stream of collections of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first stream of collections of object representations include one or more instruction sets executed prior to generating the first stream of collections of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first stream of collections of object representations include one or more instruction sets executed within a threshold period of time prior to generating the first stream of collections of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first stream of collections of object representations include one or more instruction sets

executed subsequent to generating the first stream of collections of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first stream of collections of object representations include one or more instruction sets executed within a threshold period of time subsequent to generating the first stream of collections of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first stream of collections of object representations include one or more instruction sets executed within a threshold period of time prior to generating the first stream of collections of object representations or a threshold period of time subsequent to generating the first stream of collections of object representations.

In certain embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device include a knowledge of how the device operated in a circumstance. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device are included in a neuron, a node, a vertex, or an element of a knowledgebase. In further embodiments, the knowledgebase includes a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. In further embodiments, some of the neurons, nodes, vertices, or elements are interconnected. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device are structured into a knowledge cell. In further embodiments, the knowledge cell is included in a neuron, a node, a vertex, or an element of a knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device includes correlating the first stream of collections of object representations with the first one or more instruction sets for operating the device. In further embodiments, the correlating the first stream of collections of object representations with the first one or more instruction sets for operating the device includes generating a knowledge cell, the knowledge cell comprising the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device. In further embodiments, the correlating the first stream of collections of object representations with the first one or more instruction sets for operating the device includes structuring a knowledge of how the device operated in a circumstance. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device includes learning a user's knowledge, style, or methodology of operating the device in a circumstance.

In some embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device includes storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device into the memory unit, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device being part of a plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device stored in the memory unit. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device are organized into a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, each stream of collections of object representations correlated with one or more instruction sets for operating the device of the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device is included in a neuron, a node, a vertex, or an element of a knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device include a user's knowledge, style, or methodology of operating the device in circumstances. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device are stored on a remote computing device or a remote computing system. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device include an artificial intelligence system for knowledge structuring, storing, or representation. The artificial intelligence system for knowledge structuring, storing, or representation may include at least one of: a deep learning system, a supervised learning system, an unsupervised learning system, a neural network, a search-based system, an optimization-based system, a logic-based system, a fuzzy logic-based system, a tree-based system, a graph-based system, a hierarchical system, a symbolic system, a sub-symbolic system, an evolutionary system, a genetic system, a multi-agent system, a deterministic system, a probabilistic system, or a statistical system.

In certain embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes comparing at least one portion of the new stream of collections of object representations with at least one portion of the first stream of collections of object representations. In further embodiments, the at least one portion of the new stream of collections of object representations include at least one collection of object representations, at least one object representation, or at least one object property of the new stream of collections of object representations. In further embodiments, the at least one portion of the first stream of collections of object representations include at least one collection of object representations, at least one object representation, or at least one object property of the first stream of collections of object representations. In further embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object

representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes comparing at least one collection of object representations from the new stream of collections of object representations with at least one collection of object representations from the first stream of collections of object representations. In further embodiments, the comparing at least one collection of object representations from the new stream of collections of object representations with at least one collection of object representations from the first stream of collections of object representations includes comparing at least one object representation of the at least one collection of object representations from the new stream of collections of object representations with at least one object representation of the at least one collection of object representations from the first stream of collections of object representations. The comparing at least one object representation of the at least one collection of object representations from the new stream of collections of object representations with at least one object representation of the at least one collection of object representations from the first stream of collections of object representations may include comparing at least one object property of the at least one object representation of the at least one collection of object representations from the new stream of collections of object representations with at least one object property of the at least one object representation of the at least one collection of object representations from the first stream of collections of object representations.

In some embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that there is at least a partial match between one or more portions of the new stream of collections of object representations and one or more portions of the first stream of collections of object representations. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that a similarity between at least one portion of the new stream of collections of object representations and at least one portion of the first stream of collections of object representations exceeds a similarity threshold. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining a substantial similarity between at least one portion of the new stream of collections of object representations and at least one portion of the first stream of collections of object representations. The substantial similarity may be achieved when a similarity between the at least one portion of the new stream of collections of object representations and the at least one portion of the first stream of collections of object representations exceeds a similarity threshold. The substantial similarity may be achieved when a number or a percentage of matching or partially matching portions of the new stream of collections of object representations and portions of the first stream of collections of object representations exceeds a threshold number or threshold percentage. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that a number or a percentage of matching or partially matching collections of object representations from the new stream of collections of object representations and from the first stream of collections of object representations exceeds a threshold number or threshold percentage. The matching or partially matching collections of object representations from the new stream of collections of object representations and from the first stream of collections of object representations may be determined factoring in at least one of: an importance of a collection of object representations, an order of a collection of object representations, a threshold for a similarity in a collection of object representations, or a threshold for a difference in a collection of object representations. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that a number or a percentage of matching or partially matching object representations from the new stream of collections of object representations and from the first stream of collections of object representations exceeds a threshold number or threshold percentage. The matching or partially matching object representations from the new stream of collections of object representations and from the first stream of collections of object representations may be determined factoring in at least one of: an association of an object representation with a collection of object representations, a type of an object representation, an importance of an object representation, a threshold for a similarity in an object representation, or a threshold for a difference in an object representation. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that a number or a percentage of matching or partially matching object properties from the new stream of collections of object representations and from the first stream of collections of object representations exceeds a threshold number or threshold percentage. The matching or partially matching object properties from the new stream of collections of object representations and from the first stream of collections of object representations may be determined factoring in at least one of: an association of an object property with an object representation, an association of an object property with a collection of object representations, a category of an object property, an importance of an object property, a threshold for a similarity in an object property, or a threshold for a difference in an object property. In further embodiments, determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that there is at least a partial match between at least one collection of object representations from the new stream of collections of object representations and at least one collection of object representations from the first stream of collections of object representations. The determining that there is at least a partial match between at least one collection of object representations from the new stream of collections of object representations and at least one collection of object representations from the first stream of collections of object

representations may include determining that there is at least a partial match between at least one object representation of the at least one collection of object representations from the new stream of collections of object representations and at least one object representation of the at least one collection of object representations from the first stream of collections of object representations. The determining that there is at least a partial match between at least one object representation of the at least one collection of object representations from the new stream of collections of object representations and at least one object representation of the at least one collection of object representations from the first stream of collections of object representations may include determining that there is at least a partial match between at least one object property of the at least one object representation of the at least one collection of object representations from the new stream of collections of object representations and at least one object property of the at least one object representation of the at least one collection of object representations from the first stream of collections of object representations.

In some embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations instead of or prior to an instruction set that would have been executed next. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations into a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting the processor circuit to the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting the processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes transmitting, to the processor circuit for execution, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes issuing an interrupt to the processor circuit and executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations following the interrupt. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying an element that is part of, operating on, or coupled to the processor circuit.

In certain embodiments, the processor circuit includes a logic circuit, and wherein the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the logic circuit includes a microcontroller. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying an element of the logic circuit. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations into an element of the logic circuit. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting the logic circuit to the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes replacing inputs into the logic circuit with the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes replacing outputs from the logic circuit with the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In some embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes causing an application for operating the device to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the application running on the processor circuit.

In certain embodiments, the system further comprises: an application including instruction sets for operating the device, the application running on the processor circuit, wherein the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying the application.

In some embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating

the device correlated with the first stream of collections of object representations includes redirecting an application to the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting an application to one or more alternate instruction sets, the application running on the processor circuit, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying at least one of: the memory unit, a register of the processor circuit, a storage, or a repository where instruction sets are stored or used. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets for operating an application or an object of the application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying at least one of: an element of the processor circuit, an element of the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing an assembly language. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes adding or inserting additional code into a code of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes at least one of: modifying, removing, rewriting, or overwriting a code of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of an application, the application running on the processor circuit. The branching or redirecting the code may include inserting at least one of: a branch, a jump, or a means for redirecting an execution. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes implementing a user's knowledge, style, or methodology of operating the device in a circumstance.

In certain embodiments, the system further comprises: an interface configured to cause execution of instruction sets, wherein the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations is caused by the interface. The interface may include a modification interface.

In some embodiments, the performing the one or more operations defined by the first one or more instruction sets

for operating the device correlated with the first stream of collections of object representations includes implementing a user's knowledge, style, or methodology of operating the device in a circumstance.

In certain embodiments, the system further comprises: an application running on the processor circuit.

In some embodiments, the instruction sets for operating the device are part of an application for operating the device, the application running on the processor circuit.

In certain embodiments, the system further comprises: an application for operating the device, the application running on the processor circuit. The application for operating the device may include the instruction sets for operating the device.

In some embodiments, the artificial intelligence unit is further configured to: receive at least one extra information. In further embodiments, the at least one extra information include one or more of: a time information, a location information, a computed information, or a contextual information. In further embodiments, the at least one extra information include one or more of: an information on an object, an information on an object representation, an information on a collection of object representations, an information on a stream of collections of object representations, an information on a device's circumstance, an information on an instruction set, an information on an application, an information on the processor circuit, an information on the device, or an information on an user. In further embodiments, the artificial intelligence unit is further configured to: learn the first stream of collections of object representations correlated with the at least one extra information. The learning the first stream of collections of object representations correlated with at least one extra information may include correlating the first stream of collections of object representations with the at least one extra information. The learning the first stream of collections of object representations correlated with at least one extra information may include storing the first stream of collections of object representations correlated with the at least one extra information into the memory unit. In further embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations. The anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations may include comparing an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations. The anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between an extra information correlated with the new stream of collections of object representations and

an extra information correlated with the first stream of collections of object representations may include determining that a similarity between an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations exceeds a similarity threshold.

In certain embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: cause the user interface to present a user with an option to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In some embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: receive, via the user interface, a user's selection to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In certain embodiments, the artificial intelligence unit is further configured to: rate the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. The rating the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations may include causing a user interface to display the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations along with one or more rating values as options to be selected by a user. The rating the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations may include rating the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations without a user input.

In some embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: cause the user interface to present a user with an option to cancel the execution of the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the canceling the execution of the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes restoring the processor circuit or the device to a prior state. The restoring the processor circuit or the device to a prior state may include saving the state of the processor circuit or the device prior to executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In certain embodiments, the system further comprises: an input device configured to receive a user's operating directions, the user's operating directions for instructing the processor circuit on how to operate the device. In some embodiments, the autonomous device operating includes a partially or a fully autonomous device operating. The partially autonomous device operating may include executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations responsive to a user confirmation. The fully autonomous device operating may include executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations without a user confirmation.

In certain embodiments, the artificial intelligence unit is further configured to: receive a second stream of collections of object representations, the second stream of collections of object representations including one or more collections of representations of objects detected by the sensor; receive a second one or more instruction sets for operating the device; and learn the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device. In further embodiments, the second stream of collections of object representations includes one or more collections of representations of objects detected by the sensor over time. In further embodiments, the second stream of collections of object representations includes or is associated with a time stamp, an order, or a time related information. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device include creating a connection between the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device. The connection may include or is associated with at least one of: an occurrence count, a weight, a parameter, or a data. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device include updating a connection between the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device. In further embodiments, the updating the connection between the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device includes updating at least one of: an occurrence count, a weight, a parameter, or a data included in or associated with the connection. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device includes storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device into a first node of a knowledgebase, and wherein the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device includes storing the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device into a second node of the knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. In further embodiments, the knowledgebase may be stored in the memory unit. The learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device may include creating a connection between the first node and the second node. The learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device may include updating a connection between the first node and the second node. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device is stored into a first node of a neural network and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device is stored into a second node of the neural network. The first node and the second node may be connected by a connection. The first node may be part of a first layer of the neural network and the second node may be part of a second layer of the neural network. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device is stored into a first node of a graph and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device is stored into a second node of the graph. The first node and the second node may be connected by a connection. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device is stored into a first node of a sequence and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device is stored into a second node of the sequence.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first stream of collections of object representations, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The operations may further comprise: receiving a first one or more instruction sets for operating a device. The operations may further comprise: learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device. The operations may further comprise: receiving a new stream of collections of object representations, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The operations may further comprise: anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The operations may further comprise: causing an execution of the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, wherein the device

performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the execution.

In certain embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from the one or more processor circuits or from another one or more processor circuits. In further embodiments, the execution of the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations is performed by the one or more processor circuits or by another one or more processor circuits.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first stream of collections of object representations by a processor circuit, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more instruction sets for operating a device by the processor circuit. The method may further comprise: (c) learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the processor circuit. The method may further comprise: (d) receiving a new stream of collections of object representations by the processor circuit, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The method may further comprise: (e) anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, the anticipating of (e) performed by the processor circuit. The method may further comprise: (f) executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the executing of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the executing of (f).

In some embodiments, the receiving of (b) includes receiving the first one or more instruction sets for operating the device from the processor circuit or from another processor circuit. In further embodiments, the executing of (f) is performed by the processor circuit or by another processor circuit.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable as well as the following embodiments.

In certain embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device includes storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device into a memory unit, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device being part of a plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device stored in the memory unit.

In some embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations instead of or prior to an instruction set that would have been executed next. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying a register or an element of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations into a register or an element of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting a processor circuit to the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting a processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes transmitting, to a processor circuit for execution, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes issuing an interrupt to a processor circuit and executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations following the interrupt. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying an element that is part of, operating on, or coupled to a processor circuit.

In certain embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes executing, by a logic circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the logic circuit includes a microcontroller. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the

device correlated with the first stream of collections of object representations includes modifying an element of the logic circuit. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations into an element of the logic circuit. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting the logic circuit to the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes replacing inputs into the logic circuit with the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes replacing outputs from the logic circuit with the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In some embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes executing, by an application for operating the device, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying an application, the application including instruction sets for operating the device. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting an application to the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting an application to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying at least one of: the memory unit, a register of a processor circuit, a storage, or a repository where instruction sets are stored or used. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets for operating an application or an object of the application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying at least one of: an element of a processor circuit, an element of the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing an assembly language. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes adding or inserting additional code into a code of an application. In

further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes at least one of: modifying, removing, rewriting, or overwriting a code of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of an application. The branching or redirecting the code may include inserting at least one of: a branch, a jump, or a means for redirecting an execution. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes implementing a user's knowledge, style, or methodology of operating the device in a circumstance. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations via an interface. The interface may include a modification interface.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving at least one extra information. In further embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: learning the first stream of collections of object representations correlated with the at least one extra information.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: presenting, via a user interface, a user with an option to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving, via a user interface, a user's selection to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: rating the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: presenting, via a user interface, a user with an option to cancel the execution of the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving, via an input device, a user's operating directions, the user's operating directions for instructing a processor circuit on how to operate the device.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving a second stream of collections of object representations, the second stream of collections of object representations including one or more collections of representations of objects detected by the sensor; receiving a second one or more instruction sets for operating the device; and learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving, by a first processor circuit of the one or more processor circuits, a first stream of collections of object representations, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The operations may further comprise: receiving, by the first processor circuit of the one or more processor circuits, a first one or more instruction sets for operating a device. The operations may further comprise: learning, by the first processor circuit of the one or more processor circuits, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device. The operations may further comprise: receiving, by the first processor circuit of the one or more processor circuits, a new stream of collections of object representations, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The operations may further comprise: anticipating, by the first processor circuit of the one or more processor circuits, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The operations may further comprise: causing, by the first processor circuit of the one or more processor circuits, an execution, by a second processor circuit of the one or more processor circuits, of the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the causing performed in response to the anticipating, by the first processor circuit of the one or more processor circuits, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the execution.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first stream of collections of object representations by a first processor circuit, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more instruction sets for operating a device by the first processor circuit. The method may further comprise: (c) learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the first processor circuit. The method may further comprise: (d) receiving a new stream of collections of object representations by the first processor circuit, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The method may further comprise: (e) anticipating the first one or more instruction sets for operating the device

correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, the anticipating of (e) performed by the first processor circuit. The method may further comprise: (f) executing, by a second processor circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the executing of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the executing of (f).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for learning a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first stream of collections of object representations, the first stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: learn the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first stream of collections of object representations, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The operations may further comprise: receiving a first one or more instruction sets for operating a device. The operations may further comprise: learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a non method comprising: (a) receiving a first stream of collections of object representations by a processor circuit, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more instruction sets for operating a device by the processor circuit. The method may further comprise: (c) learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the processor circuit.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: access the memory unit that comprises a plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device including a first stream of collections of object representations correlated with a first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: receive a new stream of collections of object representations, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations performed in response to the executing by the processor circuit.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: accessing a memory unit that comprises a plurality of streams of collections of object representations correlated with one or more instruction sets for operating a device, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device including a first stream of collections of object representations correlated with a first one or more instruction sets for operating the device. The operations may further comprise: receiving a new stream of collections of object representations, the new stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The opera-

tions may further comprise: anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The operations may further comprise: causing an execution of the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the execution.

In some aspects, the disclosure relates to a method comprising: (a) accessing a memory unit that comprises a plurality of streams of collections of object representations correlated with one or more instruction sets for operating a device, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device including a first stream of collections of object representations correlated with a first one or more instruction sets for operating the device, the accessing of (a) performed by a processor circuit. The method may further comprise: (b) receiving a new stream of collections of object representations by the processor circuit, the new stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The method may further comprise: (c) anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, the anticipating of (c) performed by the processor circuit. The method may further comprise: (d) executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the executing of (d) performed in response to the anticipating of (c). The method may further comprise: (e) performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the executing of (d).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for learning and using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a logic circuit configured to receive inputs and produce outputs, wherein the outputs are used for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first collection of object representations, the first collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more inputs, wherein the first one or more inputs are also received by the logic circuit. The artificial intelligence unit may be further configured to: learn the first collection of object representations correlated with the first one or more inputs. The artificial intelligence unit may be further configured to: receive a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may be further configured to: cause the logic circuit to receive the first one or more inputs correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by one or more outputs for operating the device produced by the logic circuit.

In some embodiments, the logic circuit configured to receive inputs and produce outputs includes a logic circuit configured to produce outputs based at least in part on logic operations performed on the inputs. In further embodiments, the learning the first collection of object representations correlated with the first one or more inputs includes correlating the first collection of object representations with the first one or more inputs. In further embodiments, the learning the first collection of object representations correlated with the first one or more inputs includes storing the first collection of object representations correlated with the first one or more inputs into the memory unit, the first collection of object representations correlated with the first one or more inputs being part of a plurality of collections of object representations correlated with one or more inputs stored in the memory unit. In further embodiments, the anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes comparing at least one portion of the new collection of object representations with at least one portion of the first collection of object representations. In further embodiments, the anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes determining that there is at least a partial match between the new collection of object representations and the first collection of object representations. In further embodiments, the causing the logic circuit to receive the first one or more inputs correlated with the first collection of object representations includes transmitting, to the logic circuit, the first one or more inputs correlated with the first collection of object representations. In further embodiments, the causing the logic circuit to receive the first one or more inputs correlated with the first collection of object representations includes replacing one or more inputs into the logic circuit

with the first one or more inputs correlated with the first collection of object representations.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first collection of object representations, the first collection of object representations including one or more representations of objects detected by a sensor. The operations may further comprise: receiving a first one or more inputs, wherein the first one or more inputs are also received by a logic circuit, and wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device. The operations may further comprise: learning the first collection of object representations correlated with the first one or more inputs. The operations may further comprise: receiving a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The operations may further comprise: anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further comprise: causing the logic circuit to receive the first one or more inputs correlated with the first collection of object representations, the causing performed in response to the anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, wherein the device performs one or more operations defined by one or more outputs for operating the device produced by the logic circuit.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a processor circuit, the first collection of object representations including one or more representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more inputs by the processor circuit, wherein the first one or more inputs are also received by a logic circuit, and wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device. The method may further comprise: (c) learning the first collection of object representations correlated with the first one or more inputs, the learning of (c) performed by the processor circuit. The method may further comprise: (d) receiving a new collection of object representations by the processor circuit, the new collection of object representations including one or more representations of objects detected by the sensor. The method may further comprise: (e) anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (e) performed by the processor circuit. The method may further comprise: (f) receiving, by the logic circuit, the first one or more inputs correlated with the first collection of object representations, the receiving of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the device, one or more operations defined by one or more outputs for operating the device produced by the logic circuit.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by

any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for learning and using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a logic circuit configured to receive inputs and produce outputs, wherein the outputs are used for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence. The artificial intelligence unit may be configured to: receive a first collection of object representations, the first collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more outputs, the first one or more outputs transmitted from the logic circuit. The artificial intelligence unit may be further configured to: learn the first collection of object representations correlated with the first one or more outputs. The artificial intelligence unit may be further configured to: receive a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more outputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may be further configured to: cause the device to perform one or more operations defined by the first one or more outputs correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit.

In some embodiments, the learning the first collection of object representations correlated with the first one or more outputs includes correlating the first collection of object representations with the first one or more outputs. In further embodiments, the learning the first collection of object representations correlated with the first one or more outputs includes storing the first collection of object representations correlated with the first one or more outputs into the memory unit, the first collection of object representations correlated with the first one or more outputs being part of a plurality of collections of object representations correlated with one or more outputs stored in the memory unit. In further embodiments, the anticipating the first one or more outputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes comparing at least one portion of the new collection of object representations with at least one portion of the first collection of object representations. In further embodiments, the anticipating the first one or more outputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes determining that there is at least a partial match between the new collection of object representations and the first collection of object representations. In further embodiments, the causing the device to perform one or more operations defined by the first one or

more outputs correlated with the first collection of object representations includes replacing one or more outputs from the logic circuit with the first one or more outputs correlated with the first collection of object representations.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first collection of object representations, the first collection of object representations including one or more representations of objects detected by a sensor. The operations may further comprise: receiving a first one or more outputs, the first one or more outputs transmitted from a logic circuit, wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device. The operations may further comprise: learning the first collection of object representations correlated with the first one or more outputs. The operations may further comprise: receiving a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The operations may further comprise: anticipating the first one or more outputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further comprise: causing the device to perform one or more operations defined by the first one or more outputs correlated with the first collection of object representations, the causing performed in response to the anticipating the first one or more outputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a processor circuit, the first collection of object representations including one or more representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more outputs by the processor circuit, the first one or more outputs transmitted from a logic circuit, wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device. The method may further comprise: (c) learning the first collection of object representations correlated with the first one or more outputs, the learning of (c) performed by the processor circuit. The method may further comprise: (d) receiving a new collection of object representations by the processor circuit, the new collection of object representations including one or more representations of objects detected by the sensor. The method may further comprise: (e) anticipating the first one or more outputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (e) performed by the processor circuit. The method may further comprise: (f) performing, by the device, one or more operations defined by the first one or more outputs correlated with the first collection of object representations, the one or more operations by the device performed in response to the anticipating of (e).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for learning and using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: an actuator configured to receive inputs and perform motions. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first collection of object representations, the first collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more inputs, wherein the first one or more inputs are also received by the actuator. The artificial intelligence unit may be further configured to: learn the first collection of object representations correlated with the first one or more inputs. The artificial intelligence unit may be further configured to: receive a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may be further configured to: cause the actuator to receive the first one or more inputs correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the actuator performs one or more motions defined by the first one or more inputs correlated with the first collection of object representations.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first collection of object representations, the first collection of object representations including one or more representations of objects detected by a sensor. The operations may further comprise: receiving a first one or more inputs, wherein the first one or more inputs are also received by an actuator, and wherein the actuator is configured to receive inputs and perform motions. The operations may further comprise: learning the first collection of object representations correlated with the first one or more inputs. The operations may further comprise: receiving a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The operations may further comprise: anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further comprise: causing the actuator to receive the first one or more inputs correlated with the first collection of object representations, the causing performed in response to the anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the

new collection of object representations and the first collection of object representations, wherein the actuator performs one or more motions defined by the first one or more inputs correlated with the first collection of object representations.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a processor circuit, the first collection of object representations including one or more representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more inputs by the processor circuit, wherein the first one or more inputs are also received by an actuator, and wherein the actuator is configured to receive inputs and perform motions. The method may further comprise: (c) learning the first collection of object representations correlated with the first one or more inputs, the learning of (c) performed by the processor circuit. The method may further comprise: (d) receiving a new collection of object representations by the processor circuit, the new collection of object representations including one or more representations of objects detected by the sensor. The method may further comprise: (e) anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (e) performed by the processor circuit. The method may further comprise: (f) receiving, by the actuator, the first one or more inputs correlated with the first collection of object representations, the receiving of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the actuator, one or more motions defined by the first one or more inputs correlated with the first collection of object representations.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

Other features and advantages of the disclosure will become apparent from the following description, including the claims and drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** illustrates a block diagram of Computing Device **70** that can provide processing capabilities used in some of the disclosed embodiments.

FIG. **2** illustrates an embodiment of Device **98** comprising Unit for Learning and/or Using a Device's Circumstances for Autonomous Device Operation (DCADO Unit **100**).

FIGS. **3**A-**3**E illustrate various embodiments of Sensors **92** and elements of Object Processing Unit **93**.

FIGS. **4**A-**4**B, illustrate an exemplary embodiment of Objects **615** detected in Device's **98** surrounding, and resulting Collection of Object Representations **525**.

FIG. **5** illustrates some embodiments of obtaining instruction sets, data, and/or other information through tracing, profiling, or sampling of Processor **11** registers, memory, or other computing system components.

FIGS. **6**A-**6**B illustrate some embodiments of obtaining instruction sets, data, and/or other information through tracing, profiling, or sampling of Logic Circuit **250**.

FIGS. **7**A-**7**E illustrate some embodiments of Instruction Sets **526**.

FIGS. **8**A-**8**B illustrate some embodiments of Extra Information **527**.

FIG. **9** illustrates an embodiment where DCADO Unit **100** is part of or operating on Processor **11**.

FIG. **10** illustrates an embodiment where DCADO Unit **100** resides on Server **96** accessible over Network **95**.

FIG. **11** illustrates an embodiment of learning and/or using Remote Device's **97** circumstances for autonomous Remote Device **97** operation.

FIG. **12** illustrates an embodiment of Artificial Intelligence Unit **110**.

FIG. **13** illustrates an embodiment of Knowledge Structuring Unit **520** correlating individual Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527**.

FIG. **14** illustrates another embodiment of Knowledge Structuring Unit **520** correlating individual Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527**.

FIG. **15** illustrates an embodiment of Knowledge Structuring Unit **520** correlating streams of Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527**.

FIG. **16** illustrates another embodiment of Knowledge Structuring Unit **520** correlating streams of Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527**.

FIG. **17** illustrates various artificial intelligence methods, systems, and/or models that can be utilized in DCADO Unit **100** embodiments.

FIG. **18**A-**18**C illustrate embodiments of interconnected Knowledge Cells **800** and updating weights of Connections **853**.

FIG. **19** illustrates an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Collection of Knowledge Cells **530***d*.

FIG. **20** illustrates an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Neural Network **530***a*.

FIG. **21** illustrates an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Neural Network **530***a* comprising shortcut Connections **853**.

FIG. **22** illustrates an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Graph **530***b*.

FIG. **23** illustrates an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Collection of Sequences **530***c*.

FIG. **24** illustrates an embodiment of determining anticipatory Instruction Sets **526** from a single Knowledge Cell **800**.

FIG. **25** illustrates an embodiment of determining anticipatory Instruction Sets **526** by traversing a single Knowledge Cell **800**.

FIG. **26** illustrates an embodiment of determining anticipatory Instruction Sets **526** using collective similarity comparisons.

FIG. **27** illustrates an embodiment of determining anticipatory Instruction Sets **526** using Neural Network **530***a*.

FIG. **28** illustrates an embodiment of determining anticipatory Instruction Sets **526** using Graph **530***b*.

FIG. **29** illustrates an embodiment of determining anticipatory Instruction Sets **526** using Collection of Sequences **530***c*.

FIG. **30** illustrates some embodiments of modifying execution and/or functionality of Processor **11** through modification of Processor **11** registers, memory, or other computing system components.

FIGS. **31**A-**31**B illustrate some embodiments of modifying execution and/or functionality of Logic Circuit **250** through modification of inputs and/or outputs of Logic Circuit **250**.

FIG. **32** illustrates a flow chart diagram of an embodiment of method **9100** for learning and/or using a device's circumstances for autonomous device operation.

FIG. **33** illustrates a flow chart diagram of an embodiment of method **9200** for learning and/or using a device's circumstances for autonomous device operation.

FIG. **34** illustrates a flow chart diagram of an embodiment of method **9300** for learning and/or using a device's circumstances for autonomous device operation.

FIG. **35** illustrates a flow chart diagram of an embodiment of method **9400** for learning and/or using a device's circumstances for autonomous device operation.

FIG. **36** illustrates a flow chart diagram of an embodiment of method **9500** for learning and/or using a device's circumstances for autonomous device operation.

FIG. **37** illustrates a flow chart diagram of an embodiment of method **9600** for learning and/or using a device's circumstances for autonomous device operation.

FIG. **38** illustrates an exemplary embodiment of Loader **98***a*.

FIG. **39** illustrates an exemplary embodiment of Boat **98***b*.

FIG. **40** illustrates an exemplary embodiment of utilizing Area of Interest **450** around Boat **98***b*.

Like reference numerals in different figures indicate like elements. Horizontal or vertical " . . . " or other such indicia may be used to indicate additional instances of the same type of element n, m, x, or other such letters or indicia represent integers or other sequential numbers that follow the sequence where they are indicated. It should be noted that n, m, x, or other such letters or indicia may represent different numbers in different elements even where the elements are depicted in the same figure. In general, n, m, x, or other such letters or indicia may follow the sequence and/or context where they are indicated. Any of these or other such letters or indicia may be used interchangeably depending on context and space available. The drawings are not necessarily to scale, with emphasis instead being placed upon illustrating the embodiments, principles, and concepts of the disclosure. A line or arrow between any of the disclosed elements comprises an interface that enables the coupling, connection, and/or interaction between the elements.

DETAILED DESCRIPTION

The disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation comprise apparatuses, systems, methods, features, functionalities, and/or applications that enable learning a device's circumstances including objects with various properties along with correlated instruction sets for operating the device, storing this knowledge in a knowledgebase (i.e. neural network, graph, sequences, etc.), and operating a device autonomously. The disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, any of their elements, any of their embodiments, or a combination thereof can generally be referred to as DCADO, DCADO Unit, or as other suitable name or reference.

Referring now to FIG. **1**, an embodiment is illustrated of Computing Device **70** (also referred to simply as computing device, computing system, or other suitable name or reference, etc.) that can provide processing capabilities used in some embodiments of the forthcoming disclosure. Later described devices, systems, and methods, in combination with processing capabilities of Computing Device **70**, enable learning and/or using a device's circumstances for autonomous device operation and/or other functionalities described herein. Various embodiments of the disclosed devices, systems, and methods include hardware, functions, logic, programs, and/or a combination thereof that can be implemented using any type or form of computing, computing enabled, or other device or system such as a mobile device, a computer, a computing enabled telephone, a server, a gaming device, a television device, a digital camera, a GPS receiver, a media player, an embedded device, a supercomputer, a wearable device, an implantable device, a cloud, or any other type or form of computing, computing enabled, or other device or system capable of performing the operations described herein.

In some designs, Computing Device **70** comprises hardware, processing techniques or capabilities, programs, or a combination thereof. Computing Device **70** includes one or more central processing units, which may also be referred to as processors **11**. Processor **11** includes one or more memory ports **10** and/or one or more input-output ports, also referred to as I/O ports **15**, such as I/O ports **15**A and **15**B. Processor **11** may be special or general purpose. Computing Device **70** may further include memory **12**, which can be connected to the remainder of the components of Computing Device **70** via bus **5**. Memory **12** can be connected to processor **11** via memory port **10**. Computing Device **70** may also include display device **21** such as a monitor, projector, glasses, and/or other display device. Computing Device **70** may also include Human-machine Interface **23** such as a keyboard, a pointing device, a mouse, a touchscreen, a joystick, a remote controller, and/or other input device. In some implementations, Human-machine Interface **23** can be connected with bus **5** or directly connected with specific elements of Computing Device **70**. Computing Device **70** may include additional elements such as one or more input/output devices **13**. Processor **11** may include or be interfaced with cache memory **14**. Storage **27** may include memory, which provides an operating system **17** (i.e. also referred to as OS **17**, etc.), additional application programs **18**, and/or data space **19** in which additional data or information can be stored. Alternative memory device **16** can be connected to the remaining components of Computing Device **70** via bus **5**. Network interface **25** can also be connected with bus **5** and be used to communicate with external computing devices via a network. Some or all described elements of Computing Device **70** can be directly or operatively connected or coupled with each other using any other connection means known in art. Other additional elements may be included as needed, or some of the disclosed ones may be excluded, or a combination thereof may be utilized in alternate implementations of Computing Device **70**.

Processor **11** includes one or more circuits or devices that can execute instructions fetched from memory **12** and/or other element. Processor **11** may include any combination of hardware and/or processing techniques or capabilities for

executing or implementing logic functions or programs. Processor **11** may include a single core or a multi core processor. Processor **11** includes the functionality for loading operating system **17** and operating any application programs **18** thereon. In some embodiments, Processor **11** can be provided in a microprocessing or a processing unit, such as, for example, Snapdragon processor produced by Qualcomm Inc., processor by Intel Corporation of Mountain View, Calif., processor manufactured by Motorola Corporation of Schaumburg, Ill.; processor manufactured by Transmeta Corporation of Santa Clara, Calif.; processor manufactured by International Business Machines of White Plains, N.Y.; processor manufactured by Advanced Micro Devices of Sunnyvale, Calif., or any computing circuit or device for performing similar functions. In other embodiments, processor **11** can be provided in a graphics processing unit (GPU), visual processing unit (VPU), or other highly parallel processing circuit or device such as, for example, nVidia GeForce line of GPUs, AMD Radeon line of GPUs, and/or others. Such GPUs or other highly parallel processing circuits or devices may provide superior performance in processing operations on neural networks, graphs, and/or other data structures. In further embodiments, processor **11** can be provided in a micro controller such as, for example, Texas instruments, Atmel, Microchip Technology, ARM, Silicon Labs, Intel, and/or other lines of micro controllers. In further embodiments, processor **11** can be provided in a quantum processor such as, for example, D-Wave Systems, Microsoft, Intel, IBM, Google, Toshiba, and/or other lines of quantum processors. In further embodiments, processor **11** can be provided in a biocomputer such as DNA-based computer, protein-based computer, molecule-based computer, and/or others. In further embodiments, processor **11** includes any circuit or device for performing logic operations. Processor **11** can be based on any of the aforementioned or other available processors capable of operating as described herein. Computing Device **70** may include one or more of the aforementioned or other processors. In some designs, processor **11** can communicate with memory **12** via a system bus **5**. In other designs, processor **11** can communicate directly with memory **12** via a memory port **10**.

Memory **12** includes one or more circuits or devices capable of storing data. In some embodiments, Memory **12** can be provided in a semiconductor or electronic memory chip such as static random access memory (SRAM), Flash memory, Burst SRAM or SynchBurst SRAM (BSRAM), Dynamic random access memory (DRAM), Fast Page Mode DRAM (FPM DRAM), Enhanced DRAM (EDRAM), Extended Data Output RAM (EDO RAM), Extended Data Output DRAM (EDO DRAM), Burst Extended Data Output DRAM (BEDO DRAM), Enhanced DRAM (EDRAM), synchronous DRAM (SDRAM), JEDEC SRAM, PC100 SDRAM, Double Data Rate SDRAM (DDR SDRAM), Enhanced SDRAM (ESDRAM), SyncLink DRAM (SLDRAM), Direct Rambus DRAM (DRDRAM), Ferroelectric RAM (FRAM), and/or others. In other embodiments, Memory **12** includes any volatile memory. In general, Memory **12** can be based on any of the aforementioned or other available memories capable of operating as described herein.

Storage **27** includes one or more devices or mediums capable of storing data. In some embodiments, Storage **27** can be provided in a device or medium such as a hard drive, flash drive, optical disk, and/or others. In other embodiments, Storage **27** can be provided in a biological storage device such as DNA-based storage device, protein-based storage device, molecule-based storage device, and/or others. In further embodiments, Storage **27** can be provided in an optical storage device such as holographic storage, and/or others. In further embodiments, Storage **27** may include any non-volatile memory. In general, Storage **27** can be based on any of the aforementioned or other available storage devices or mediums capable of operating as described herein. In some aspects, Storage **27** may include any features, functionalities, and embodiments of Memory **12**, and vice versa, as applicable.

Processor **11** can communicate directly with cache memory **14** via a connection means such as a secondary bus which may also sometimes be referred to as a backside bus. In some embodiments, processor **11** can communicate with cache memory **14** using the system bus **5**. Cache memory **14** may typically have a faster response time than main memory **12** and can include a type of memory which is considered faster than main memory **12** such as, for example, SRAM, BSRAM, or EDRAM. Cache memory includes any structure such as multilevel caches, for example. In some embodiments, processor **11** can communicate with one or more I/O devices **13** via a system bus **5**. Various busses can be used to connect processor **11** to any of the I/O devices **13** such as a VESA VL bus, an ISA bus, an EISA bus, a MicroChannel Architecture (MCA) bus, a PCI bus, a PCI-X bus, a PCI-Express bus, a NuBus, and/or others. In some embodiments, processor **11** can communicate directly with I/O device **13** via HyperTransport, Rapid I/O, or InfiniBand. In further embodiments, local busses and direct communication can be mixed. For example, processor **11** can communicate with an I/O device **13** using a local interconnect bus and communicate with another I/O device **13** directly. Similar configurations can be used for any other components described herein.

Computing Device **70** may further include alternative memory such as a SD memory slot, a USB memory stick, an optical drive such as a CD-ROM drive, a CD-R/RW drive, a DVD-ROM drive or a BlueRay disc, a hard-drive, and/or any other device comprising non-volatile memory suitable for storing data or installing application programs. Computing Device **70** may further include a storage device **27** comprising any type or form of non-volatile memory for storing an operating system (OS) such as any type or form of Windows OS, Mac OS, Unix OS, Linux OS, Android OS, iPhone OS, mobile version of Windows OS, an embedded OS, or any other OS that can operate on Computing Device **70**. Computing Device **70** may also include application programs **18**, and/or data space **19** for storing additional data or information. In some embodiments, alternative memory **16** can be used as or similar to storage device **27**. Additionally, OS **17** and/or application programs **18** can be operable from a bootable medium such as, for example, a flash drive, a micro SD card, a bootable CD or DVD, and/or other bootable medium.

Application Program **18** (also referred to as program, computer program, application, script, code, or other suitable name or reference) comprises instructions that can provide functionality when executed by processor **11**. As such, Application Program **18** may be used to operate (i.e. perform operations on/with) or control a device or system. Application program **18** can be implemented in a high-level procedural or object-oriented programming language, or in a low-level machine or assembly language. Any language used can be compiled, interpreted, or otherwise translated into machine language. Application program **18** can be deployed in any form including as a stand-alone program or as a module, component, subroutine, or other unit suitable for use in a computing system. Application program **18** does not necessarily correspond to a file in a file system. A

program can be stored in a portion of a file that may hold other programs or data, in a single file dedicated to the program, or in multiple files (i.e. files that store one or more modules, sub programs, or portions of code, etc.). Application Program **18** can be delivered in various forms such as, for example, executable file, library, script, plugin, addon, applet, interface, console application, web application, application service provider (ASP)-type application, operating system, and/or other forms. Application program **18** can be deployed to be executed on one computing device or on multiple computing devices (i.e. cloud, distributed, or parallel computing, etc.), or at one site or distributed across multiple sites interconnected by a communication network or an interface.

Network interface **25** can be utilized for interfacing Computing Device **70** with other devices via a network through a variety of connections including telephone lines, wired or wireless connections, LAN or WAN links (i.e. 802.11, T1, T3, 56 kb, X.25, etc.), broadband connections (i.e. ISDN, Frame Relay, ATM, etc.), or a combination thereof. Examples of networks include the Internet, an intranet, an extranet, a local area network (LAN), a wide area network (WAN), a personal area network (PAN), a home area network (HAN), a campus area network (CAN), a metropolitan area network (MAN), a global area network (GAN), a storage area network (SAN), virtual network, a virtual private network (VPN), a Bluetooth network, a wireless network, a wireless LAN, a radio network, a HomePNA, a power line communication network, a G.hn network, an optical fiber network, an Ethernet network, an active networking network, a client-server network, a peer-to-peer network, a bus network, a star network, a ring network, a mesh network, a star-bus network, a tree network, a hierarchical topology network, and/or other networks. Network interface **25** may include a built-in network adapter, network interface card, PCMCIA network card, card bus network adapter, wireless network adapter, Bluetooth network adapter, WiFi network adapter, USB network adapter, modem, and/or any other device suitable for interfacing Computing Device **70** with any type of network capable of communication and/or operations described herein.

I/O devices **13** may be present in various shapes or forms in Computing Device **70**. Examples of I/O device **13** capable of input include a joystick, a keyboard, a mouse, a trackpad, a trackpoint, a touchscreen, a trackball, a microphone, a drawing tablet, a glove, a tactile input device, a still or video camera, and/or other input device. Examples of I/O device **13** capable of output include a video display, a touchscreen, a projector, a glasses, a speaker, a tactile output device, and/or other output device. Examples of I/O device **13** capable of input and output include a disk drive, an optical storage device, a modem, a network card, and/or other input/output device. I/O device **13** can be interfaced with processor **11** via an I/O port **15**, for example. In some aspects, I/O device **13** can be a bridge between system bus **5** and an external communication bus such as a USB bus, an Apple Desktop Bus, an RS-232 serial connection, a SCSI bus, a FireWire bus, a FireWire 800 bus, an Ethernet bus, an AppleTalk bus, a Gigabit Ethernet bus, an Asynchronous Transfer Mode bus, a HIPPI bus, a Super HIPPI bus, a SerialPlus bus, a SCI/LAMP bus, a FibreChannel bus, a Serial Attached small computer system interface bus, and/or other bus.

An output interface (not shown) such as a graphical user interface, an acoustic output interface, a tactile output interface, any device driver (i.e. audio, video, or other driver), and/or other output interface or system can be utilized to process output from elements of Computing Device **70** for conveyance on an output device such as Display **21**. In some aspects, Display **21** or other output device itself may include an output interface for processing output from elements of Computing Device **70**. Further, an input interface (not shown) such as a keyboard listener, a touchscreen listener, a mouse listener, any device driver (i.e. audio, video, keyboard, mouse, touchscreen, or other driver), and/or other input interface or system can be utilized to process input from Human-machine Interface **23** or other input device for use by elements of Computing Device **70**. In some aspects, Human-machine Interface **23** or other input device itself may include an input interface for processing input for use by elements of Computing Device **70**.

Computing Device **70** may include or be connected to multiple display devices **21**. Display devices **21** can each be of the same or different type or form. Computing Device **70** and/or its elements comprise any type or form of suitable hardware, programs, or a combination thereof to support, enable, or provide for the connection and use of multiple display devices **21**. In one example, Computing Device **70** includes any type or form of video adapter, video card, driver, and/or library to interface, communicate, connect, or otherwise use display devices **21**. In some aspects, a video adapter may include multiple connectors to interface to multiple display devices **21**. In other aspects, Computing Device **70** includes multiple video adapters, with each video adapter connected to one or more display devices **21**. In some embodiments, Computing Device's **70** operating system can be configured for using multiple displays **21**. In other embodiments, one or more display devices **21** can be provided by one or more other computing devices such as remote computing devices connected to Computing Device **70** via a network or an interface.

Computing Device **70** can operate under the control of operating system **17**, which may support Computing Device's **70** basic functions, interface with and manage hardware resources, interface with and manage peripherals, provide common services for application programs, schedule tasks, and/or perform other functionalities. A modern operating system enables features and functionalities such as a high resolution display, graphical user interface (GUI), touchscreen, cellular network connectivity (i.e. mobile operating system, etc.), Bluetooth connectivity, WiFi connectivity, global positioning system (GPS) capabilities, mobile navigation, microphone, speaker, still picture camera, video camera, voice recorder, speech recognition, music player, video player, near field communication, personal digital assistant (PDA), and/or other features, functionalities, or applications. For example, Computing Device **70** can use any conventional operating system, any embedded operating system, any real-time operating system, any open source operating system, any video gaming operating system, any proprietary operating system, any online operating system, any operating system for mobile computing devices, or any other operating system capable of running on Computing Device **70** and performing operations described herein. Example of operating systems include Windows XP, Windows 7, Windows 8, Windows 10, etc. manufactured by Microsoft Corporation of Redmond, Wash.; Mac OS, iPhone OS, etc. manufactured by Apple Computer of Cupertino, Calif.; OS/2 manufactured by International Business Machines of Armonk, N.Y.; Linux, a freely-available operating system distributed by Caldera Corp. of Salt Lake City, Utah; or any type or form of a Unix operating system, and/or

others. Any operating systems such as the ones for Android devices can similarly be utilized.

Computing Device **70** can be implemented as or be part of various model architectures such as web services, distributed computing, grid computing, cloud computing, and/or other architectures. For example, in addition to the traditional desktop, server, or mobile operating system architectures, a cloud-based operating system can be utilized to provide the structure on which embodiments of the disclosure can be implemented. Other aspects of Computing Device **70** can also be implemented in the cloud without departing from the spirit and scope of the disclosure. For example, memory, storage, processing, and/or other elements can be hosted in the cloud. In some embodiments, Computing Device **70** can be implemented on multiple devices. For example, a portion of Computing Device **70** can be implemented on a mobile device and another portion can be implemented on wearable electronics.

Computing Device **70** can be or include any mobile device, a mobile phone, a smartphone (i.e. iPhone, Windows phone, Blackberry phone, Android phone, etc.), a tablet, a personal digital assistant (PDA), wearable electronics, implantable electronics, and/or other mobile device capable of implementing the functionalities described herein. Computing Device **70** can also be or include an embedded device, which can be any device or system with a dedicated function within another device or system. Embedded systems range from the simplest ones dedicated to one task with no user interface to complex ones with advanced user interface that may resemble modern desktop computer systems. Examples of devices comprising an embedded device include a mobile telephone, a personal digital assistant (PDA), a gaming device, a media player, a digital still or video camera, a pager, a television device, a set-top box, a personal navigation device, a global positioning system (GPS) receiver, a portable storage device (i.e. a USB flash drive, etc.), a digital watch, a DVD player, a printer, a microwave oven, a washing machine, a dishwasher, a gateway, a router, a hub, an automobile entertainment system, an automobile navigation system, a refrigerator, a washing machine, a factory automation device, an assembly line device, a factory floor monitoring device, a thermostat, an automobile, a factory controller, a telephone, a network bridge, and/or other devices. An embedded device can operate under the control of an operating system for embedded devices such as MicroC/OS-II, QNX, VxWorks, eCos, TinyOS, Windows Embedded, Embedded Linux, and/or other embedded device operating systems.

Various implementations of the disclosed devices, systems, and methods can be realized in digital electronic circuitry, integrated circuitry, logic gates, application specific integrated circuits (ASICs), field programmable gate arrays (FPGAs), computer hardware, firmware, programs, virtual machines, and/or combinations thereof including their structural, logical, and/or physical equivalents.

The disclosed devices, systems, and methods may include clients and servers. A client and server are generally, but not always, remote from each other and typically, but not always, interact via a network or an interface.

The relationship of a client and server may arise by virtue of computer programs running on their respective computers and having a client-server relationship to each other, for example.

The disclosed devices, systems, and methods can be implemented in a computing system that includes a back end component, a middleware component, a front end component, or any combination thereof. The components of the system can be interconnected by any form or medium of digital data communication such as, for example, a network.

Computing Device **70** may include or be interfaced with a computer program product comprising instructions or logic encoded on a computer-readable medium. Such instructions or logic, when executed, may configure or cause one or more processors to perform the operations and/or functionalities disclosed herein. For example, a computer program can be provided or encoded on a computer-readable medium such as an optical medium (i.e. DVD-ROM, etc.), flash drive, hard drive, any memory, firmware, or other medium. Computer program can be installed onto a computing device to cause the computing device to perform the operations and/or functionalities disclosed herein. Machine-readable medium, computer-readable medium, or other such terms may refer to any computer program product, apparatus, and/or device for providing instructions and/or data to one or more programmable processors. As such, machine-readable medium includes any medium that can send and/or receive machine instructions as a machine-readable signal. Examples of a machine-readable medium include a volatile and/or non-volatile medium, a removable and/or non-removable medium, a communication medium, a storage medium, and/or other medium. A communication medium, for example, can transmit computer readable instructions and/or data in a modulated data signal such as a carrier wave or other transport technique, and may include any other form of information delivery medium known in art. A non-transitory machine-readable medium comprises all machine-readable media except for a transitory, propagating signal.

In some embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, can be implemented entirely or in part in a device (i.e. microchip, circuitry, logic gates, electronic device, computing device, special or general purpose processor, etc.) or system that comprises (i.e. hard coded, internally stored, etc.) or is provided with (i.e. externally stored, etc.) instructions for implementing DCADO functionalities. As such, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, may include the processing, memory, storage, and/or other features, functionalities, and embodiments of Computing Device **70** or elements thereof. Such device or system can operate on its own (i.e. standalone device or system, etc.), be embedded in another device or system (i.e. an industrial machine, a robot, a vehicle, a toy, a smartphone, a television device, an appliance, and/or any other device or system capable of housing the elements needed for DCADO functionalities), work in combination with other devices or systems, or be available in any other configuration. In other embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, may include Alternative Memory **16** that provides instructions for implementing DCADO functionalities to one or more Processors **11**. In further embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, can be implemented entirely or in part as a computer program and executed by one or more Processors **11**. Such program can be implemented in one or more modules or units of a single or multiple computer programs. Such program may be able to attach to or interface with, inspect, and/or take control of another application program

to implement DCADO functionalities. In further embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, can be implemented as a network, web, distributed, cloud, or other such application accessed on one or more remote computing devices (i.e. servers, cloud, etc.) via Network Interface **25**, such remote computing devices including processing capabilities and instructions for implementing DCADO functionalities. In further embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, can be (1) attached to or interfaced with any computing device or application program, (2) included as a feature of an operating system, (3) built (i.e. hard coded, etc.) into any computing device or application program, and/or (4) available in any other configuration to provide its functionalities.

In some embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, can be implemented at least in part in a computer program such as Java application or program. Java provides a robust and flexible environment for application programs including flexible user interfaces, robust security, built-in network protocols, powerful application programming interfaces, database or DBMS connectivity and interfacing functionalities, file manipulation capabilities, support for networked applications, and/or other features or functionalities. Application programs based on Java can be portable across many devices, yet leverage each device's native capabilities. Java supports the feature sets of most smartphones and a broad range of connected devices while still fitting within their resource constraints. Various Java platforms include virtual machine features comprising a runtime environment for application programs. Java platforms provide a wide range of user-level functionalities that can be implemented in application programs such as displaying text and graphics, playing and recording audio content, displaying and recording visual content, communicating with another computing device, and/or other functionalities. It should be understood that the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, are programming language, platform, and operating system independent. Examples of programming languages that can be used instead of or in addition to Java include C, C++, Cobol, Python, Java Script, Tcl, Visual Basic, Pascal, VB Script, Perl, PHP, Ruby, and/or other programming languages capable of implementing the functionalities described herein.

Where a reference to a specific file or file type is used herein, other files or file types can be substituted.

Where a reference to a data structure is used herein, it should be understood that any variety of data structures can be used such as, for example, array, list, linked list, doubly linked list, queue, tree, heap, graph, map, grid, matrix, multi-dimensional matrix, table, database, database management system (DBMS), file, neural network, and/or any other type or form of a data structure including a custom one. A data structure may include one or more fields or data fields that are part of or associated with the data structure. A field or data field may include a data, an object, a data structure, and/or any other element or a reference/pointer thereto. A data structure can be stored in one or more memories, files, or other repositories. A data structure and/or elements thereof, when stored in a memory, file, or other repository, may be stored in a different arrangement than the arrangement of the data structure and/or elements thereof. For example, a sequence of elements can be stored in an arrangement other than a sequence in a memory, file, or other repository.

Where a reference to a repository is used herein, it should be understood that a repository may be or include one or more files or file systems, one or more storage locations or structures, one or more storage systems, one or more memory locations or structures, and/or other file, storage, memory, or data arrangements.

Where a reference to an interface is used herein, it should be understood that the interface comprises any hardware, device, system, program, method, and/or combination thereof that enable direct or operative coupling, connection, and/or interaction of the elements between which the interface is indicated. A line or arrow shown in the figures between any of the depicted elements comprises such interface. Examples of an interface include a direct connection, an operative connection, a wired connection (i.e. wire, cable, etc.), a wireless connection, a device, a network, a bus, a circuit, a firmware, a driver, a bridge, a program, a combination thereof, and/or others.

Where a reference to an element coupled or connected to another element is used herein, it should be understood that the element may be in communication or other interactive relationship with the other element. Furthermore, an element coupled or connected to another element can be coupled or connected to any other element in alternate implementations. Terms coupled, connected, interfaced, or other such terms may be used interchangeably herein depending on context.

Where a reference to an element matching another element is used herein, it should be understood that the element may be equivalent or similar to the other element. Therefore, the term match or matching can refer to total equivalence or similarity depending on context.

Where a reference to a device is used herein, it should be understood that the device may include or be referred to as a system, and vice versa depending on context, since a device may include a system of elements and a system may be embodied in a device.

Where a reference to a collection of elements is used herein, it should be understood that the collection of elements may include one or more elements. In some aspects or contexts, a reference to a collection of elements does not imply that the collection is an element itself.

Where a reference to an object is used herein, it should be understood that the object may be a physical object (i.e. object detected in a device's surrounding, etc.), an electronic object (i.e. object in an object oriented application program, etc.), and/or other object depending on context.

Where a mention of a function, method, routine, subroutine, or other such procedure is used herein, it should be understood that the function, method, routine, subroutine, or other such procedure comprises a call, reference, or pointer to the function, method, routine, subroutine, or other such procedure.

Where a mention of data, object, data structure, item, element, or thing is used herein, it should be understood that the data, object, data structure, item, element, or thing comprises a reference or pointer to the data, object, data structure, item, element, or thing.

Referring to FIG. **2**, an embodiment of Device **98** comprising Unit for Learning and/or Using a Device's Circumstances for Autonomous Device Operation (DCADO Unit

100) is illustrated. Device **98** also comprises interconnected Processor **11**, Human-machine Interface **23**, Sensor **92**, Object Processing Unit **93**, Memory **12**, and

Storage **27**. Processor **11** includes or executes Application Program **18**. DCADO Unit **100** comprises interconnected Artificial Intelligence Unit **110**, Acquisition Interface **120**, and Modification Interface **130**. Other additional elements can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate embodiments.

In one example, the teaching presented by the disclosure can be implemented in a device or system for learning and/or using a device's circumstances for autonomous device operation. The device or system may include a processor circuit (i.e. Processor **11**, etc.) configured to execute instruction sets (i.e. Instruction Sets **526**, etc.) for operating a device. The device or system may further include a memory unit (i.e. Memory **12**, etc.) configured to store data. The device or system may further include a sensor (i.e. Sensor **92**, etc.) configured to detect objects (i.e. Objects **615**, etc.). The device or system may further include an artificial intelligence unit (i.e. Artificial Intelligence Unit **110**, etc.). The artificial intelligence unit may be configured to receive a first collection of object representations (i.e. Collection of Object Representations **525**, etc.), the first collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may also be configured to receive a first one or more instruction sets for operating the device. The artificial intelligence unit may also be configured to learn the first collection of object representations correlated with the first one or more instruction sets for operating the device. The artificial intelligence unit may also be configured to receive a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may also be configured to anticipate the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may also be configured to cause the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations performed in response to the executing by the processor circuit. Any of the operations of the aforementioned elements can be performed repeatedly and/or in different orders in alternate embodiments. In some embodiments, a stream of collections of object representations can be used instead of or in addition to any collection of object representations such as, for example, using a first stream of collections of object representations instead of the first collection of object representations. In other embodiments, a logic circuit (i.e. Logic Circuit **250**, etc.) may be used instead of the processor circuit. In such embodiments, one or more instruction sets for operating the device (i.e. first one or more instruction sets for operating the device, etc.) may include or be substituted with one or more inputs into or one or more outputs from the logic circuit. In further embodiments, an actuator may be included instead of or in addition to the processor circuit. In such embodiments, one or more instruction sets for operating the device (i.e. first one or more instruction sets for operating the device, etc.) may include or be substituted with one or more inputs into the actuator. Other additional elements can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate embodiments. The device or system for learning and/or using a device's circumstances for autonomous device operation may include any actions or operations of any of the disclosed methods such as methods **9100**, **9200**, **9300**, **9400**, **9500**, **9600**, and/or others (all later described).

Device **98** comprises any hardware, programs, or a combination thereof. Although, Device **98** is referred to as a device herein, Device **98** may be or include a system as a system may be embodied in Device **98**. Device **98** may include any features, functionalities, and embodiments of Computing Device **70**, or elements thereof. In some embodiments, Device **98** includes a computing enabled device for performing mechanical or physical operations (i.e. via actuators, etc.). In other embodiments, Device **98** includes a computing enabled device for performing non-mechanical and/or other operations. Examples of Device **98** include an industrial machine, a toy, a robot, a vehicle, an appliance, a control device, a smartphone or other mobile computer, any computer, and/or other computing enabled device or machine. Such device or machine may be built for any function or purpose some examples of which are described later.

User **50** (also referred to simply as user or other suitable name or reference) comprises a human user or non-human user. A non-human User **50** includes any device, system, program, and/or other mechanism for operating or controlling Device **98** and/or elements thereof. In one example, User **50** may issue an operating direction to Application Program **18** responsive to which Application Program's **18** instructions or instruction sets may be executed by Processor **11** to perform a desired operation on Device **98**. In another example, User **50** may issue an operating direction to Processor **11**, Logic Circuit **250** (later described), and/or other processing element responsive to which Processor **11**, Logic Circuit **250**, and/or other processing element may implement logic to perform a desired operation on Device **98**. User's **50** operating directions comprise any user inputted data (i.e. values, text, symbols, etc.), directions (i.e. move right, move up, move forward, copy an item, click on a link, etc.), instructions or instruction sets (i.e. manually inputted instructions or instruction sets, etc.), and/or other inputs or information. A non-human User **50** can utilize more suitable interfaces instead of, or in addition to, Human-machine Interface **23** and/or Display **21** for controlling Device **98** and/or elements thereof. Examples of such interfaces include an application programming interface (API), bridge (i.e. bridge between applications, devices, or systems, etc.), driver, socket, direct or operative connection, handle, function/routine/subroutine, and/or other interfaces.

In some embodiments, Processor **11**, Logic Circuit **250**, Application Program **18**, and/or other processing element may control or affect an actuator (not shown). Actuator comprises the functionality for implementing motion, actions, behaviors, maneuvers, and/or other mechanical or physical operations. Device **98** may include one or more actuators to enable Device **98** to perform mechanical, physical, or other operations and/or to interact with its environment. For example, an actuator may include or be coupled to an element such as a wheel, arm, or other element to act upon the environment. Examples of an actuator include a motor, a linear motor, a servomotor, a hydraulic element, a

pneumatic element, an electro-magnetic element, a spring element, and/or other actuators. Examples of types of actuators include a rotary actuator, a linear actuator, and/or other types of actuators. In other embodiments, Processor **11**, Logic Circuit **250**, Application Program **18**, and/or other processing element may control or affect any other device or element instead of or in addition to an actuator.

Referring to FIGS. **3**A-**3**E, various embodiments of Sensors **92** and elements of Object Processing Unit **93** are illustrated.

Sensor **92** (also referred to as simply as sensor or other suitable name or reference) comprises the functionality for obtaining or detecting information about its environment, and/or other functionalities. As such, one or more Sensors **92** can be used to detect objects and/or their properties in Device's **98** surrounding. In some aspects, Device's **98** surrounding may include exterior of Device **98**. In other aspects, Device's **98** surrounding may include interior of Device **98** in case of hollow Device **98**, Device **98** comprising compartments or openings, and/or other variously shaped Device **98**. Examples of aspects of an environment that Sensor **92** can measure or be sensitive to include light (i.e. camera, lidar, etc.), electromagnetism/electromagnetic field (i.e. radar, etc.), sound (i.e. microphone, sonar, etc.), physical contact (i.e. tactile sensor, etc.), magnetism/magnetic field (i.e. compass, etc.), electricity/electric field, temperature, gravity, vibration, pressure, and/or others. In some aspects, a passive sensor (i.e. camera, microphone, etc.) measures signals or radiation emitted or reflected by an object. In other aspects, an active sensor (i.e. lidar, radar, sonar, etc.) emits signals or radiation and measures the signals or radiation reflected or backscattered from an object. A reference to a Sensor **92** herein includes a reference to one or more Sensors **92** as applicable. In some designs, a plurality of Sensors **92** may be used to detect objects and/or their properties from different angles or sides of Device **98**. For example, four Cameras **92***a* can be placed on four corners of Device **98** to cover 360 degrees of view of Device's **98** surrounding. In other designs, a plurality of different types of Sensors **92** may be used to detect different types of objects and/or their properties. For example, one or more Cameras **92***a* can be used to detect and identify an object, whereas, Radar **92***d* can be used to determine distance and bearing/angle of the object relative to Device **98**. In further designs, a signal-emitting element can be placed within or onto an object and Sensor **92** can detect the signal from the signal-emitting element, thereby detecting the object and/or its properties. For example, a radio-frequency identification (RFID) emitter may be placed within an object to help Sensor **92** detect, identify, and/or obtain other information about the object.

In some embodiments, Sensor **92** may be or include Camera **92***a* as shown in FIG. **3**A. Camera **92***a* comprises the functionality for capturing one or more pictures, and/or other functionalities. As such, Camera **92***a* can be used to capture pictures of Device's **98** surrounding. Camera **92***a* may be useful in detecting existence of an object, type of an object, identity of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In some aspects, Camera **92***a* may be or comprises a motion picture camera that can capture streams of pictures (i.e. motion pictures, videos, etc.). In other aspects, Camera **92***a* may be or comprises a still picture camera that can capture still pictures (i.e. photographs, etc.). In further aspects, Camera **92***a* may be or comprises a stereo camera (i.e. camera with multiple lenses, etc.) that can capture stereoscopic or range pictures. In further aspects, Camera **92***a* may be or comprises any other Camera **92***a*. In general, Camera **92***a* may capture any light (i.e. visible light, infrared light, ultraviolet light, x-ray light, etc.) across the electromagnetic spectrum onto a light-sensitive material. Any other technique known in art can be utilized to facilitate Camera **92***a* functionalities. In one example, a digital Camera **92***a* can utilize a charge coupled device (CCD), a complementary metal-oxide-semiconductor (CMOS) sensor, and/or other electronic image sensor to capture digital pictures that can then be stored in a memory or storage, or transmitted to any of the disclosed or other elements for further processing. In another example, analog Camera **92***a* can utilize an analog-to-digital converter to produce digital pictures. In some embodiments, Camera **92***a* can be built, embedded, or integrated in Device **98** and/or other disclosed element. In other embodiments, Camera **92***a* can be an external Camera **92***a* connected with Device **98** and/or other disclosed element. In further embodiments, Camera **92***a* comprises Computing Device **70** or elements thereof. In general, Camera **92***a* can be implemented in any suitable configuration to provide its functionalities. Camera **92***a* may capture one or more digital pictures. A digital picture may include a collection of color encoded pixels or dots. Examples of file formats that can be utilized to store a digital picture include JPEG, GIF, TIFF, PNG, PDF, and/or other digitally encoded picture formats. A stream of digital pictures (i.e. motion picture, video, etc.) may include one or more digital pictures. Examples of file formats that can be utilized to store a stream of digital pictures include MPEG, AVI, FLV, MOV, RM, SWF, WMV, DivX, and/or other digitally encoded motion picture formats.

In other embodiments, Sensor **92** may be or include Microphone **92***b* as shown in FIG. **3**B. Microphone **92***b* comprises the functionality for capturing one or more sounds, and/or other functionalities. As such, Microphone **92***b* can be used to capture sounds from Device's **98** surrounding. Microphone **92***b* may be useful in detecting existence of an object, type of an object, identity of an object, bearing/angle of an object, activity (i.e. motion, sounding, etc.) of an object, and/or other properties of an object. In some aspects, Microphone **92***b* may be omnidirectional microphone that enables capturing sounds from any direction. In other aspects, Microphone **92***b* may be a directional (i.e. unidirectional, bidirectional, etc.) microphone that enables capturing sounds from one or more directions while ignoring or being insensitive to sounds from other directions. In general, Microphone **92***b* may utilize a membrane sensitive to air pressure and may produce electrical signal from air pressure variations. Samples of the electrical signal can then be read to produce a stream of digital sound samples. Any other technique known in art can be utilized to facilitate Microphone **92***b* functionalities. In one example, a digital Microphone **92***b* may include an integrated analog-to-digital converter to capture a stream of digital sound samples that can then be stored in a memory or storage, or transmitted to any of the disclosed or other elements for further processing. In another example, analog Microphone **92***b* may utilize an external analog-to-digital converter to produce a stream of digital sound samples. In some embodiments, Microphone **92***b* can be built, embedded, or integrated in Device **98**. In other embodiments, Microphone **92***b* can be an external Microphone **92***b* connected with Device **98**. In further embodiments where used in water, Microphone **92***b* may be or include a hydrophone. In further embodiments, Microphone **92***b* comprises Computing Device **70** or elements thereof. In general, Micro-

phone **92***b* can be implemented in any suitable configuration to provide its functionalities. Examples of file formats that can be utilized to store a stream of digital sound samples include WAV, WMA, AIFF, MP3, RA, OGG, and/or other digitally encoded sound formats.

In further embodiments, Sensor **92** may be or include Lidar **92***c* as shown in FIG. **3**C. Lidar **92***c* may be useful in detecting existence of an object, type of an object, identity of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In some aspects, Lidar **92***c* may emit a light signal (i.e. laser beam, etc.) and listen for a signal that is reflected or backscattered from an object. Any other technique known in art can be utilized to facilitate Lidar **92***c* functionalities.

In further embodiments, Sensor **92** may be or include a Radar **92***d* as shown in FIG. **3**D. Radar **92***d* may be useful in detecting existence of an object, type of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In some aspects, Radar **92***d* may emit a radio signal (i.e. radio wave, etc.) and listen for a signal that is reflected or backscattered from an object. Any other technique known in art can be utilized to facilitate Radar **92***d* functionalities.

In further embodiments, Sensor **92** may be or include Sonar **92***e* as shown in FIG. **3**E. Sonar **92***e* may be useful in detecting existence of an object, type of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In some aspects, Sonar **92***e* may emit a sound signal (i.e. sound pulse, etc.) and listen for a signal that is reflected or backscattered from an object. Any other technique known in art can be utilized to facilitate Sonar **92***e* functionalities.

One of ordinary skill in art will understand that the aforementioned sensors are described merely as examples of a variety of possible implementations, and that while all possible sensors are too voluminous to describe, other sensors known in art that can facilitate detecting of objects and/or their properties in Device's **98** surrounding are within the scope of this disclosure. Any combination of the aforementioned and/or other sensors can be used in various embodiments.

Object Processing Unit **93** comprises the functionality for processing output from Sensor **92** to obtain information of interest, and/or other functionalities. As such, Object Processing Unit **93** can be used to process output from Sensor **92** to detect objects and/or their properties in Device's **98** surrounding. In some embodiments, Object Processing Unit **93** comprises the functionality for creating or generating Collection of Object Representations **525** (also referred to as Coll of Obj Rep or other suitable name or reference) and storing one or more Object Representations **625** (also referred to simply as object representations, representations of objects, or other suitable name or reference), Object Properties **630** (also referred to simply as object properties or other suitable name or reference), and/or other elements or information into the Collection of Object Representations **525**. As such, Collection of Object Representations **525** comprises the functionality for storing one or more Object Representations **625**, Object Properties **630**, and/or other elements or information. Object Representation **625** may include an electronic representation of an object (i.e. Object **615** [later described], etc.) detected in Device's **98** surrounding. In some aspects, Collection of Object Representations **525** includes one or more Object Representations **625**, Object Properties **630**, and/or other elements or information related to objects detected in Device's **98** surrounding at a particular time. Collection of Object Representations **525** may, therefore, include knowledge (i.e. unit of knowledge, etc.) of Device's **98** circumstances including objects with various properties at a particular time. In some designs, a Collection of Object Representations **525** may include or be associated with a time stamp (not shown), order (not shown), or other time related information. For example, one Collection of Object Representations **525** may be associated with time stamp t**1**, another Collection of Object Representations **525** may be associated with time stamp t**2**, and so on. Time stamps t**1**, t**2**, etc. may indicate the times of generating Collections of Object Representations **525**, for instance. In other embodiments, Object Processing Unit **93** comprises the functionality for creating or generating a stream of Collections of Object Representations **525**. A stream of Collections of Object Representations **525** may include one Collection of Object Representations **525** or a group, sequence, or other plurality of Collections of Object Representations **525**. In some aspects, a stream of Collections of Object Representations **525** includes one or more Collections of Object Representations **525**, and/or other elements or information related to objects detected in Device's **98** surrounding over time. A stream of Collections of Object Representations **525** may, therefore, include knowledge (i.e. unit of knowledge, etc.) of Device's **98** circumstances including objects with various properties over time. As circumstances including objects with various properties in Device's **98** surrounding change (i.e. objects and/or their properties change, move, act, transform, etc.) over time, this change may be captured in a stream of Collections of Object Representations **525**. In some designs, each Collection of Object Representations **525** in a stream may include or be associated with the aforementioned time stamp, order, or other time related information. For example, one Collection of Object Representations **525** in a stream may be associated with order **1**, a next Collection of Object Representations **525** in the stream may be associated with order **2**, and so on. Orders **1**, **2**, etc. may indicate the orders or places of Collections of Object Representations **525** within a stream (i.e. sequence, etc.), for instance. Examples of objects include biological objects (i.e. persons, animals, vegetation, etc.), nature objects (i.e. rocks, bodies of water, etc.), manmade objects (i.e. buildings, streets, ground/aerial/aquatic vehicles, etc.), and/or others. In some aspects, any part of an object may be detected as an object itself. For instance, instead of or in addition to detecting a vehicle as an object, a wheel and/or other parts of the vehicle may be detected as objects. In general, object may include any object or part thereof that can be detected. Examples of object properties include existence of an object, type of an object (i.e. person, cat, vehicle, building, street, tree, rock, etc.), identity of an object (i.e. name, identifier, etc.), distance of an object, bearing/angle of an object, location of an object (i.e. distance and bearing/angle from a known point, coordinates, etc.), shape/size of an object (i.e. height, width, depth, computer model, point cloud, etc.), activity of an object (i.e. motion, gestures, etc.), and/or other properties of an object. Type of an object, for example, may include any classification of objects ranging from detailed such as person, cat, vehicle, building, street, tree, rock, etc. to generalized such as biological object, nature object, manmade object, etc., and/or others including their sub-types. Location of an object, for example, can include a relative location such as one defined by distance and bearing/angle from a known point or location (i.e. Device **98** location, etc.). Location of an object, for

example, can also include absolute location such as one defined by object coordinates. In general, an object property may include any attribute of an object (i.e. existence of an object, type of an object, identity of an object, shape/size of an object, etc.), any relationship of an object with Device **98**, other objects, or the environment (i.e. distance of an object, bearing/angle of an object, friend/foe relationship, etc.), and/or other information related to an object. In some implementations, Object Processing Unit **93** and/or any of its elements or functionalities can be included in Sensor **92**. In other implementations, Object Processing Unit **93** and/or any of its elements or functionalities can be embedded into or operate on Processor **11**. In further implementations, Object Processing Unit **93** and/or any of its elements or functionalities can be embedded into or operate in DCADO Unit **100**, and/or other disclosed elements. Object Processing Unit **93** may be provided in any suitable configuration. Object Processing Unit **93** may include any signal processing techniques or elements known in art as applicable.

In some embodiments, Object Processing Unit **93** may include Picture Recognizer **94***a* as shown in FIG. **3**A. Picture Recognizer **94***a* comprises the functionality for detecting or recognizing objects and/or their properties in visual data, and/or other disclosed functionalities. Visual data includes digital motion pictures, digital still pictures, and/or other visual data. Examples of file formats that can be utilized to store visual data include AVI, DivX, MPEG, JPEG, GIF, TIFF, PNG, PDF, and/or other file formats. For example, Picture Recognizer **94***a* can be used for detecting or recognizing objects and/or their properties in one or more digital pictures captured by one or more Cameras **92***a*. Picture Recognizer **94***a* can be utilized in detecting or recognizing existence of an object, type of an object, identity of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In general, Picture Recognizer **94***a* can be used for any operation supported by Picture Recognizer **94***a*. Picture Recognizer **94***a* may detect or recognize an object and/or its properties as well as track the object and/or its properties in one or more digital pictures or streams of digital pictures (i.e. motion pictures, video, etc.). In the case of a person, Picture Recognizer **94***a* may detect or recognize a human head or face, upper body, full body, or portions/combinations thereof. In some aspects, Picture Recognizer **94***a* may detect or recognize objects and/or their properties from a digital picture by comparing regions of pixels from the digital picture with collections of pixels comprising known objects and/or their properties. The collections of pixels comprising known objects and/or their properties can be learned or manually, programmatically, or otherwise defined. The collections of pixels comprising known objects and/or their properties can be stored in any data structure or repository (i.e. one or more files, database, etc.) that resides locally on Device **98**, or remotely on a remote computing device (i.e. server, cloud, etc.) accessible over a network or an interface. In other aspects, Picture Recognizer **94***a* may detect or recognize objects and/or their properties from a digital picture by comparing features (i.e. lines, edges, ridges, corners, blobs, regions, etc.) of the digital picture with features of known objects and/or their properties. The features of known objects and/or their properties can be learned or manually, programmatically, or otherwise defined. The features of known objects and/or their properties can be stored in any data structure or repository (i.e. neural network, one or more files, database, etc.) that resides locally on Device **98**, or remotely on a remote computing device (i.e. server, cloud, etc.) accessible over a network or an interface. Typical steps or elements in a feature oriented picture recognition include pre-processing, feature extraction, detection/segmentation, decision-making, and/or others, or a combination thereof, each of which may include its own sub-steps or sub-elements depending on the application. In further aspects, Picture Recognizer **94***a* may detect or recognize multiple objects and/or their properties from a digital picture using the aforementioned pixel or feature comparisons, and/or other detection or recognition techniques. For example, a picture may depict two objects in two of its regions both of which Picture Recognizer **94***a* can detect simultaneously. In further aspects, where objects and/or their properties span multiple pictures, Picture Recognizer **94***a* may detect or recognize objects and/or their properties by applying the aforementioned pixel or feature comparisons and/or other detection or recognition techniques over a stream of digital pictures (i.e. motion picture, video, etc.). For example, once an object is detected in a digital picture (i.e. frame, etc.) of a stream of digital pictures (i.e. motion picture, video, etc.), the region of pixels comprising the detected object or the object's features can be searched in other pictures of the stream of digital pictures, thereby tracking the object through the stream of digital pictures. In further aspects, Picture Recognizer **94***a* may detect or recognize an object's activities by identifying and/or analyzing differences between a detected region of pixels of one picture (i.e. frame, etc.) and detected regions of pixels of other pictures in a stream of digital pictures. For example, a region of pixels comprising a person's face can be detected in multiple consecutive pictures of a stream of digital pictures (i.e. motion picture, video, etc.). Differences among the detected regions of the consecutive pictures may be identified in the mouth part of the person's face to indicate smiling or speaking activity. In further aspects, Picture Recognizer **94***a* may detect or recognize objects and/or their properties using one or more artificial neural networks, which may include statistical techniques. Examples of artificial neural networks that can be used in Picture Recognizer **94***a* include convolutional neural networks (CNNs), time delay neural networks (TDNNs), deep neural networks, and/or others. In one example, picture recognition techniques and/or tools involving convolutional neural networks may include identifying and/or analyzing tiled and/or overlapping regions or features of a digital picture, which may then be used to search for pictures with matching regions or features. In another example, features of different convolutional neural networks responsible for spatial and temporal streams can be fused to detect objects and/or their properties in streams of digital pictures (i.e. motion pictures, videos, etc.). In general, Picture Recognizer **94***a* may include any machine learning, deep learning, and/or other artificial intelligence techniques. In further aspects, Picture Recognizer **94***a* can detect distance of a recognized object in a picture captured by a camera using structured light, sheet of light, or other lighting schemes, and/or by using phase shift analysis, time of flight, interferometry, or other techniques. In further aspects, Picture Recognizer **94***a* may detect distance of a recognized object in a picture captured by a stereo camera by using triangulation and/or other techniques. In further aspects, Picture Recognizer **94***a* may detect bearing/angle of a recognized object relative to the camera-facing direction by measuring the distance from the vertical centerline of the picture to a pixel in the recognized object based on known picture resolution and camera's angle of view. Any other techniques known in art can be utilized in Picture Recognizer **94***a*. For example, thresholds for similarity, statistical

techniques, and/or optimization techniques can be utilized to determine a match in any of the above-described detection or recognition techniques. In some exemplary embodiments, object recognition techniques and/or tools such as OpenCV (Open Source Computer Vision) library, CamFind API, Kooaba, 6px API, Dextro API, and/or others can be utilized for detecting or recognizing objects and/or their properties in digital pictures. In some aspects, picture recognition techniques and/or tools involve identifying and/or analyzing features such as lines, edges, ridges, corners, blobs, regions, and/or their relative positions, sizes, shapes, etc., which may then be used to search for pictures with matching features. For example, OpenCV library can detect an object (i.e. person, animal, vehicle, rock, etc.) and/or its properties in one or more digital pictures captured by Camera **92***a* or stored in an electronic repository, which can then be utilized in DCADO Unit **100**, Artificial Intelligence Unit **110**, and/or other elements. In other exemplary embodiments, facial recognition techniques and/or tools such as OpenCV (Open Source Computer Vision) library, Animetrics FaceR API, Lambda Labs Facial Recognition API, Face++ SDK, Neven Vision (also known as N-Vision) Engine, and/or others can be utilized for detecting or recognizing faces in digital pictures. In some aspects, facial recognition techniques and/or tools involve identifying and/or analyzing facial features such as the relative position, size, and/or shape of the eyes, nose, cheekbones, jaw, etc., which may then be used to search for pictures with matching features. For example, FaceR API can detect a person's face in one or more digital pictures captured by Camera **92***a* or stored in an electronic repository, which can then be utilized in DCADO Unit **100**, Artificial Intelligence Unit **110**, and/or other elements.

Various aspects or properties of digital pictures or pixels can be taken into account by Picture Recognizer **94***a* in any of the recognizing or comparisons. Examples of such aspects or properties include color adjustment, size adjustment, content manipulation, transparency (i.e. alpha channel, etc.), use of mask, and/or others. In some implementations, as digital pictures can be captured by various picture taking equipment, in various environments, and under various lighting conditions, Picture Recognizer **94***a* can adjust lighting or color of pixels or otherwise manipulate pixels before or during comparison. Lighting or color adjustment (also referred to as gray balance, neutral balance, white balance, etc.) may generally include manipulating or rebalancing the intensities of the colors (i.e. red, green, and/or blue if RGB color model is used, etc.) of one or more pixels. For example, Picture Recognizer **94***a* can adjust lighting or color of some or all pixels of one picture to make it more comparable to another picture. Picture Recognizer **94***a* can also incrementally adjust the pixels such as increasing or decreasing the red, green, and/or blue pixel values by a certain amount in each cycle of comparisons in order to find a substantially similar match at one of the incremental adjustment levels. Any of the publically available, custom, or other lighting or color adjustment techniques or programs can be utilized such as color filters, color balancing, color correction, and/or others. In other implementations, Picture Recognizer **94***a* can resize or otherwise transform a digital picture before or during comparison. Such resizing or transformation may include increasing or decreasing the number of pixels of a digital picture. For example, Picture Recognizer **94***a* can increase or decrease the size of a digital picture proportionally (i.e. increase or decrease length and/or width keeping aspect ratio constant, etc.) to equate its size with the size of another digital picture. Picture Recognizer **94***a* can also incrementally resize a digital picture such as increasing or decreasing the size of the digital picture proportionally by a certain amount in each cycle of comparisons in order to find a substantially similar match at one of the incremental sizes. Any of the publically available, custom, or other digital picture resizing techniques or programs can be utilized such as nearest-neighbor interpolation, bilinear interpolation, bicubic interpolation, and/or others. In further implementations, Picture Recognizer **94***a* can manipulate content (i.e. all pixels, one or more regions, one or more depicted objects, etc.) of a digital picture before or during comparison. Such content manipulation may include moving, centering, aligning, resizing, transforming, and/or otherwise manipulating content of a digital picture. For example, Picture Recognizer **94***a* can move, center, or align content of one picture to make it more comparable to another picture. Any of the publically available, custom, or other digital picture manipulation techniques or programs can be utilized such as pixel moving, warping, distorting, aforementioned interpolations, and/or others. In further implementations, in digital pictures comprising transparency features or functionalities, Picture Recognizer **94***a* can utilize a threshold for acceptable number or percentage transparency difference. Alternatively, transparency can be applied to one or more pixels of a digital picture and color difference may then be determined between compared pixels taking into account the transparency related color effect. Alternatively, transparent pixels can be excluded from comparison. In further implementations, certain regions or subsets of pixels can be ignored or excluded during comparison using a mask. In general, any region or subset of a picture determined to contain no content of interest can be excluded from comparison using a mask. Examples of such regions or subsets include background, transparent or partially transparent regions, regions comprising insignificant content, or any arbitrary region or subset. Picture Recognizer **94***a* can perform any other pre-processing or manipulation of digital pictures or pixels before or during recognizing or comparison.

In other embodiments, Object Processing Unit **93** may include Sound Recognizer **94***b* as shown in FIG. **3**B. Sound Recognizer **94***b* comprises the functionality for detecting or recognizing objects and/or their properties in audio data, and/or other disclosed functionalities. Audio data includes digital sound, and/or other audio data. Examples of file formats that can be utilized to store audio data include WAV, WMA, AIFF, MP3, RA, OGG, and/or other file formats. For example, Sound Recognizer **94***b* can be used for detecting or recognizing objects and/or their properties in a stream of digital sound samples captured by one or more Microphones **92***b*. In the case of a person, Sound Recognizer **94***b* may detect or recognize human voice. Sound Recognizer **94***b* can be utilized in detecting or recognizing existence of an object, type of an object, identity of an object, bearing/angle of an object, activity (i.e. motion, sounding, etc.) of an object, and/or other properties of an object. In general, Sound Recognizer **94***b* can be used for any operation supported by Sound Recognizer **94***b*. In some aspects, Sound Recognizer **94***b* may detect or recognize an object and/or its properties from a stream of digital sound samples by comparing collections of sound samples from the stream of digital sound samples with collections of sound samples of known objects and/or their properties. The collections of sound samples of known objects and/or their properties can be learned, or manually, programmatically, or otherwise defined. The collections of sound samples of known objects and/or their properties can be stored in any data structure or

repository (i.e. one or more files, database, etc.) that resides locally on Device **98**, or remotely on a remote computing device (i.e. server, cloud, etc.) accessible over a network or an interface. In other aspects, Sound Recognizer **94***b* may detect or recognize an object and/or its properties from a stream of digital sound samples by comparing features from the stream of digital sound samples with features of sounds of known objects and/or their properties. The features of sounds of known objects and/or their properties can be learned, or manually, programmatically, or otherwise defined. The features of sounds of known objects and/or their properties can be stored in any data structure or repository (i.e. one or more files, database, neural network, etc.) that resides locally on Device **98**, or remotely on a remote computing device (i.e. server, cloud, etc.) accessible over a network or an interface. Typical steps or elements in a feature oriented sound recognition include pre-processing, feature extraction, acoustic modeling, language modeling, and/or others, or a combination thereof, each of which may include its own sub-steps or sub-elements depending on the application. In further aspects, Sound Recognizer **94***b* may detect or recognize a variety of sounds from a stream of digital sound samples using the aforementioned sound sample or feature comparisons, and/or other detection or recognition techniques. For example, sound of a person, animal, vehicle, and/or other sounds can be detected by Sound Recognizer **94***b*. In further aspects, Sound Recognizer **94***b* may detect or recognize sounds using Hidden Markov Models (HMM), Artificial Neural Networks, Dynamic Time Warping (DTW), Gaussian Mixture Models (GMM), and/or other models or techniques, or a combination thereof. Some or all of these models or techniques may include statistical techniques. Examples of artificial neural networks that can be used in Sound Recognizer **94***b* include recurrent neural networks, time delay neural networks (TDNNs), deep neural networks, convolutional neural networks, and/or others. In general, Sound Recognizer **94***b* may include any machine learning, deep learning, and/or other artificial intelligence techniques. In further aspects, Sound Recognizer **94***b* may detect bearing/angle of a recognized object by measuring the direction in which Microphone **92***b* is pointing when sound of maximum strength is received, by analyzing amplitude of the sound, by performing phase analysis (i.e. with microphone array, etc.) of the sound, and/or by utilizing other techniques. Any other techniques known in art can be utilized in Sound Recognizer **94***b*. For example, thresholds for similarity, statistical techniques, and/or optimization techniques can be utilized to determine a match in any of the above-described detection or recognition techniques. In some exemplary embodiments, operating system's Sound recognition functionalities such as iOS's Voice Services, Siri, and/or others can be utilized in Sound Recognizer **94***b*. For example, iOS Voice Services can detect an object (i.e. person, etc.) and/or its properties in a stream of digital sound samples captured by Microphone **92***b* or stored in an electronic repository, which can then be utilized in DCADO Unit **100**, Artificial Intelligence Unit **110**, and/or other elements. In other exemplary embodiments, Java Speech API (JSAPI) implementation such as The Cloud Garden, Sphinx, and/or others can be utilized in Sound Recognizer **94***b*. For example, Cloud Garden JSAPI can detect an object (i.e. person, animal, vehicle, etc.) and/or its properties in a stream of digital sound samples captured by Microphone **92***b* or stored in an electronic repository, which can then be utilized in DCADO Unit **100**, Artificial Intelligence Unit **110**, and/or other elements. Any other programming language's or platform's speech or sound processing API can similarly be utilized. In further exemplary embodiments, applications or engines providing Sound recognition functionalities such as HTK (Hidden Markov Model Toolkit), Kaldi, OpenEars, Dragon Mobile, Julius, iSpeech, CeedVocal, and/or others can be utilized in Sound Recognizer **94***b*. For example, Kaldi SDK can detect an object (i.e. person, animal, vehicle, etc.) and/or its properties in a stream of digital sound samples captured by Microphone **92***b* or stored in an electronic repository, which can then be utilized in DCADO Unit **100**, Artificial Intelligence Unit **110**, and/or other elements.

In further embodiments, Object Processing Unit **93** may include Lidar Processing Unit **94***c* as shown in FIG. **3**C. Lidar Processing Unit **94***c* comprises the functionality for detecting or recognizing objects and/or their properties using light, and/or other disclosed functionalities. As such, Lidar Processing Unit **94***c* can be utilized in detecting existence of an object, type of an object, identity of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In general, Lidar Processing Unit **94***c* can be used for any operation supported by Lidar Processing Unit **94***c*. In one example, Lidar Processing Unit **94***c* may detect distance of an object by measuring time delay between emission of a light signal (i.e. laser beam, etc.) and return of the light signal reflected from the object based on known speed of light. In another example, Lidar Processing Unit **94***c* may detect bearing/angle of an object by analyzing the amplitudes of a light signal received by an array of detectors (i.e. detectors arranged into a quadrant or other arrangement, etc.). In a further example, Lidar Processing Unit **94***c* may detect existence, type, identity, shape/size, activity, and/or other properties of an object by illuminating the object with light and acquiring an image of the object, which can then be processed using some of the previously described or other picture recognition techniques. In a further example, Lidar Processing Unit **94***c* may detect existence, type, identity, shape/size, activity, and/or other properties of an object by illuminating the object with light and acquiring a point cloud representation of the object. Lidar Processing Unit **94***c* may detect objects and/or their properties by utilizing any lidar or light-related techniques known in art.

In further embodiments, Object Processing Unit **93** may include Radar Processing Unit **94***d* as shown in FIG. **3**D. Radar Processing Unit **94***d* comprises the functionality for detecting or recognizing objects and/or their properties using radio waves, and/or other disclosed functionalities. As such, Radar Processing Unit **94***d* can be utilized in detecting existence of an object, type of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In general, Radar Processing Unit **94***d* can be used for any operation supported by Radar Processing Unit **94***d*. In one example, Radar Processing Unit **94***d* may detect existence of an object by emitting a radio signal and listening for the radio signal reflected from the object. In another example, Radar Processing Unit **94***d* may detect distance of an object by measuring time delay between emission of a radio signal and return of the radio signal reflected from the object based on known speed of the radio signal. In a further example, Radar Processing Unit **94***d* may detect bearing/angle of an object by measuring the direction in which the antenna is pointing when the return signal of maximum strength is received, by analyzing amplitude of the return signal, by performing phase analysis (i.e. with antenna array, etc.) of the return signal, and/or by

utilizing any amplitude, phase, or other techniques. In a further example, Radar Processing Unit **94***d* may detect existence, type, identity, shape/size, activity, and/or other properties of an object by illuminating the object with radio waves and acquiring an image of the object, which can then be processed using some of the previously described or other picture recognition techniques. Radar Processing Unit **94***d* may detect objects and/or their properties by utilizing any radar or radio-related techniques known in art.

In further embodiments, Object Processing Unit **93** may include Sonar Processing Unit **94***e* as shown in FIG. **3**E. Sonar Processing Unit **94***e* comprises the functionality for detecting or recognizing objects and/or their properties using sound, and/or other disclosed functionalities. As such, Sonar Processing Unit **94***e* can be utilized in detecting existence of an object, type of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In general, Sonar Processing Unit **94***e* can be used for any operation supported by Sonar Processing Unit **94***e*. In one example, Sonar Processing Unit **94***e* may detect existence of an object by emitting a sound signal and listening for the sound signal reflected from the object. In another example, Sonar Processing Unit **94***e* may detect distance of an object by measuring time delay between emission of a sound signal and return of the sound signal reflected from the object based on known speed of the sound signal. In a further example, Sonar Processing Unit **94***e* may detect bearing/angle of an object by measuring the direction in which the microphone is pointing when the return signal of maximum strength is received, by analyzing amplitude of the return signal, by performing phase analysis (i.e. with microphone array, etc.) of the return signal, and/or by utilizing any amplitude, phase, or other techniques. In a further example, Sonar Processing Unit **94***e* may detect existence, type, identity, shape/size, activity, and/or other properties of an object by illuminating the object with sound pulses and acquiring an image of the object, which can then be processed using some of the previously described or other picture recognition techniques. Sonar Processing Unit **94***e* may detect objects and/or their properties by utilizing any sonar or sound-related techniques known in art.

One of ordinary skill in art will understand that the aforementioned techniques for detecting or recognizing objects and/or their properties are described merely as examples of a variety of possible implementations, and that while all possible techniques for detecting or recognizing objects and/or their properties are too voluminous to describe, other techniques for detecting or recognizing objects and/or their properties known in art are within the scope of this disclosure. Also, any signal processing technique known in art that can facilitate the disclosed functionalities can be utilized in various embodiments. Any combination of the aforementioned and/or other sensors, object detecting or recognizing techniques, signal processing techniques, and/or other elements or techniques can be used in various embodiments.

Referring to FIGS. **4**A-**4**B, an exemplary embodiment of Objects **615** (also referred to simply as objects or other suitable name or reference) detected in Device's **98** surrounding, and resulting Collection of Object Representations **525** are illustrated.

As shown for example in FIG. **4**A, Object **615***a* is detected. Object **615***a* may be recognized as a cat. Object **615***a* may be detected at a distance of 6 m from Device **98**. Object **615***a* may be detected at a bearing/angle of 56° from Device's **98** centerline. Furthermore, Object **615***b* is also detected. Object **615***b* may be recognized as a tree. Object **615***b* may be detected at a distance of 10 m from Device **98**. Object **615***b* may be detected at a bearing/angle of 131° from Device's **98** centerline. Furthermore, Object **615***c* is also detected. Object **615***c* may be recognized as a person. Object **615***c* may be identified as John Doe. Object **615***c* may be detected at a distance of 8 m from Device **98**. Object **615***c* may be detected at a bearing/angle of 287° from Device's **98** centerline. Any other Objects **615** instead of or in addition to Object **615***a*, Object **615***b*, and Object **615***c* may be detected. In some aspects, any features, functionalities, and embodiments of Camera **92***a*/Picture Recognizer **94***a*, Microphone **92***b*/Sound Recognizer **94***b*, and/or other sensors or techniques can be utilized for recognizing and/or identifying a person, a cat, a tree, and/or other Objects **615**. In further aspects, any features, functionalities, and embodiments of Camera **92***a*/Picture Recognizer **94***a*, Microphone **92***b*/Sound Recognizer **94***b*, Lidar **92***c*/Lidar Processing Unit **94***c*, Radar **92***d*/Radar Processing Unit **94***d*, Sonar **92***e*/Sonar Processing Unit **94***e*, and/or other sensors or techniques can be utilized for detecting distance, bearing/angle, and/or other object properties.

As shown for example in FIG. **4**B, Object Processing Unit **93** may create or generate Collection of Object Representations **525** including Object Representation **625***a* representing Object **615***a*, Object Representation **625***b* representing Object **615***b*, Object Representation **625***c* representing Object **615***c*, etc. For instance, Object Representation **625***a* may include Object Property **630***aa* "Cat" in Category **635***aa* "Type", Object Property **630***ab* "6 m" in Category **635***ab* "Distance", Object Property **630***ac* "56°" in Category **635***ac* "Bearing", etc. Also, Object Representation **625***b* may include Object Property **630***ba* "Tree" in Category **635***ba* "Type", Object Property **630***bb* "10 m" in Category **635***bb* "Distance", Object Property **630***bc* "131°" in Category **635***bc* "Bearing", etc. Also, Object Representation **625***c* may include Object Property **630***ca* "Person" in Category **635***ca* "Type", Object Property **630***cb* "John Doe" in Category **635***cb* "Identity", Object Property **630***cc* "8 m" in Category **635***cc* "Distance", Object Property **630***cd* "287°" in Category **635***cd* "Bearing", etc. Any number of Object Representations **625**, and/or other elements or information can be included in Collection of Object Representations **525**. Any number of Object Properties **630** (also referred to simply as object properties or other suitable name or reference), and/or other elements or information can be included in an Object Representation **625**. In some aspects, a reference to Collection of Object Representations **525** comprises a reference to a collection of Object Properties **630** and/or other elements or information related to one or more Objects **615**. Other additional Object Representations **625**, Object Properties **630**, elements, and/or information can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate embodiments of Collection of Object Representations **525**.

Referring now to DCADO Unit **100**, DCADO Unit **100** comprises any hardware, programs, or a combination thereof. DCADO Unit **100** comprises the functionality for learning the operation of Device **98** in circumstances including objects with various properties. DCADO Unit **100** comprises the functionality for structuring and/or storing this knowledge in a knowledgebase (i.e. neural network, graph, sequences, other repository, etc.). DCADO Unit **100** comprises the functionality for enabling autonomous operation of Device **98** in circumstances including objects with various properties. DCADO Unit **100** comprises the func-

tionality for interfacing with or attaching to Application Program **18**, Processor **11**, Logic Circuit **250** (later described), and/or other processing element. DCADO Unit **100** comprises the functionality for obtaining instruction sets, data, and/or other information used, implemented, and/or executed by Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. DCADO Unit **100** comprises the functionality for modifying instruction sets, data, and/or other information used, implemented, and/or executed by Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. DCADO Unit **100** comprises learning, anticipating, decision making, automation, and/or other functionalities disclosed herein. Statistical, artificial intelligence, machine learning, and/or other models or techniques are utilized to implement the disclosed devices, systems, and methods.

When DCADO Unit **100** functionalities are applied on Application Program **18**, Processor **11**, Logic Circuit **250** (later described), and/or other processing element of Device **98**, Device **98** may become autonomous. DCADO Unit **100** may take control from, share control with, and/or release control to Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element to implement autonomous operation of Device **98**. DCADO Unit **100** may take control from, share control with, and/or release control to Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element automatically or after prompting User **50** to allow it. In some aspects, Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element of an autonomous Device **98** may include or be provided with anticipatory (also referred to as alternate or other suitable name or reference) instructions or instruction sets that User **50** did not issue or cause to be executed. Such anticipatory instructions or instruction sets include instruction sets that User **50** may want or is likely to issue or cause to be executed. Anticipatory instructions or instruction sets can be generated by DCADO Unit **100** or elements thereof based on Device's **98** circumstances including objects with various properties. As such, Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element of an autonomous Device **98** may include or be provided with some or all original instructions or instruction sets and/or any anticipatory instructions or instruction sets generated by DCADO Unit **100**. Therefore, autonomous Device **98** operating may include executing some or all original instructions or instruction sets and/or any anticipatory instructions or instruction sets generated by DCADO Unit **100**. In one example, DCADO Unit **100** can overwrite or rewrite the original instructions or instruction sets of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element with DCADO Unit **100**-generated instructions or instruction sets. In another example, DCADO Unit **100** can insert or embed DCADO Unit **100**-generated instructions or instruction sets among the original instructions or instruction sets of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. In a further example, DCADO Unit **100** can branch, redirect, or jump to DCADO Unit **100**-generated instructions or instruction sets from the original instructions or instruction sets of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element.

In some embodiments, autonomous Device **98** operating comprises determining, by DCADO Unit **100**, a next instruction or instruction set to be executed based on Device's **98** circumstances including objects with various properties prior to the user issuing or causing to be executed the next instruction or instruction set. In yet other embodiments, autonomous application operating comprises determining, by DCADO Unit **100**, a next instruction or instruction set to be executed based on Device's **98** circumstances including objects with various properties prior to the system receiving the next instruction or instruction set.

In some embodiments, autonomous Device **98** operating includes a partially or fully autonomous operating. In an example involving partially autonomous Device **98** operating, a user confirms DCADO Unit **100**-generated instructions or instruction sets prior to their execution. In an example involving fully autonomous application operating, DCADO Unit **100**-generated instructions or instruction sets are executed without user or other system confirmation (i.e. automatically, etc.).

In some embodiments, a combination of DCADO Unit **100** and other systems and/or techniques can be utilized to implement Device's **98** operation. In one example, DCADO Unit **100** may be a primary or preferred system for implementing Device's **98** operation. While operating autonomously under the control of DCADO Unit **100**, Device **98** may encounter a circumstance including objects with various properties that has not been encountered or learned before. In such situations, User **50** and/or non-DCADO system may take control of Device's **98** operation. DCADO Unit **100** may take control again when Device **98** encounters a previously learned circumstance including objects with various properties. Naturally, DCADO Unit **100** can learn Device's **98** operation in circumstances while User **50** and/or non-DCADO system is in control of Device **98**, thereby reducing or eliminating the need for future involvement of User **50** and/or non-DCADO system. In another example, User **50** and/or non-DCADO system may be a primary or preferred system for implementing Device's **98** operation. While operating under the control of User **50** and/or non-DCADO system, User **50** and/or non-DCADO system may release control to DCADO Unit **100** for any reason (i.e. User **50** gets tired or distracted, non-DCADO system gets stuck or cannot make a decision, etc.), at which point Device **98** can be controlled by DCADO Unit **100**. In some designs, DCADO Unit **100** may take control in certain special circumstances including objects with various properties where DCADO Unit **100** may offer superior performance even though User **50** and/or non-DCADO system may generally be preferred. Once Device **98** leaves such special circumstances, DCADO Unit **100** may release control to User **50** and/or non-DCADO system. In general, DCADO Unit **100** can take control from, share control with, or release control to User **50**, non-DCADO system, and/or other system or process at any time, in any circumstances, and remain in control for any period of time as needed.

In some embodiments, DCADO Unit **100** may control one or more sub-devices, sub-systems, or elements of Device **98** while User **50** and/or non-DCADO system may control other one or more sub-devices, sub-systems, or elements of Device **98**.

It should be understood that a reference to autonomous operating of Device **98** may include autonomous operating of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element depending on context.

Referring now to Acquisition Interface **120**, Acquisition Interface **120** comprises the functionality for obtaining and/or receiving instruction sets, data, and/or other information. Acquisition Interface **120** comprises the functionality for obtaining and/or receiving instruction sets, data, and/or other information from Processor **11**, Application Program **18**, Logic Circuit **250** (later described), and/or other processing element. Acquisition Interface **120** comprises the function-

ality for obtaining and/or receiving instruction sets, data, and/or other information at runtime. In some aspects, an instruction set may include any computer command, instruction, signal, or input used in Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. Therefore, the terms instruction set, command, instruction, signal, input, or other such terms may be used interchangeably herein depending on context. Acquisition Interface **120** also comprises the functionality for attaching to or interfacing with Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. In one example, Acquisition Interface **120** comprises the functionality to access and/or read runtime engine/environment, virtual machine, operating system, compiler, just-in-time (JIT) compiler, interpreter, translator, execution stack, file, object, data structure, and/or other computing system elements. In another example, Acquisition Interface **120** comprises the functionality to access and/or read memory, storage, bus, interfaces, and/or other computing system elements. In a further example, Acquisition Interface **120** comprises the functionality to access and/or read Processor **11** registers and/or other Processor **11** elements. In a further example, Acquisition Interface **120** comprises the functionality to access and/or read inputs and/or outputs of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. In a further example, Acquisition Interface **120** comprises the functionality to access and/or read functions, methods, procedures, routines, subroutines, and/or other elements of Application Program **18**. In a further example, Acquisition Interface **120** comprises the functionality to access and/or read source code, bytecode, compiled, interpreted, or otherwise translated code, machine code, and/or other code. In a further example, Acquisition Interface **120** comprises the functionality to access and/or read values, variables, parameters, and/or other data or information. Acquisition Interface **120** also comprises the functionality for transmitting the obtained instruction sets, data, and/or other information to Artificial Intelligence Unit **110** and/or other element. As such, Acquisition Interface **120** provides input into Artificial Intelligence Unit **110** for knowledge structuring, anticipating, decision making, and/or other functionalities later in the process. Acquisition Interface **120** also comprises other disclosed functionalities.

Acquisition Interface **120** can employ various techniques for obtaining instruction sets, data, and/or other information. In one example, Acquisition Interface **120** can attach to and/or obtain Processor's **11**, Application Program's **18**, Logic Circuit's **250**, and/or other processing element's instruction sets, data, and/or other information through tracing or profiling techniques. Tracing or profiling may be used for outputting Processor's **11**, Application Program's **18**, Logic Circuit's **250**, and/or other processing element's instruction sets, data, and/or other information at runtime. For instance, tracing or profiling may include adding trace code (i.e. instrumentation, etc.) to an application and/or outputting trace information to a specific target. The outputted trace information (i.e. instruction sets, data, and/or other information, etc.) can then be provided to or recorded into a file, data structure, repository, an application, and/or other system or target that may receive such trace information. As such, Acquisition Interface **120** can utilize tracing or profiling to obtain instruction sets, data, and/or other information and provide them as input into Artificial Intelligence Unit **110**. In some aspects, instrumentation can be performed in source code, bytecode, compiled, interpreted, or otherwise translated code, machine code, and/or other code. In other aspects, instrumentation can be performed in various elements of a computing system such as memory, virtual machine, runtime engine/environment, operating system, compiler, interpreter, translator, processor registers, execution stack, program counter, and/or other elements. In yet other aspects, instrumentation can be performed in various abstraction layers of a computing system such as in software layer (i.e. Application Program **18**, etc.), in virtual machine (if VM is used), in operating system, in Processor **11**, and/or in other layers or areas that may exist in a particular computing system implementation. In yet other aspects, instrumentation can be performed at various time periods in an application's execution such as source code write time, compile time, interpretation time, translation time, linking time, loading time, runtime, and/or other time periods. In yet other aspects, instrumentation can be performed at various granularities or code segments such as some or all lines of code, some or all statements, some or all instructions or instruction sets, some or all basic blocks, some or all functions/routines/subroutines, and/or some or all other code segments.

In some embodiments, Application Program **18** can be automatically instrumented. For example, Acquisition Interface **120** can access Application Program's **18** source code, bytecode, or machine code and select instrumentation points of interest. Selecting instrumentation points may include finding locations in the source code, bytecode, or machine code corresponding to function calls, function entries, function exits, object creations, object destructions, event handler calls, new lines (i.e. to instrument all lines of code, etc.), thread creations, throws, and/or other points of interest. Instrumentation code can then be inserted at the instrumentation points of interest to output Application Program's **18** instruction sets, data, and/or other information. In response to executing instrumentation code, Application Program's **18** instruction sets, data, and/or other information may be received by Acquisition Interface **120**. In some aspects, Application Program's **18** source code, bytecode, or machine code can be dynamically instrumented. For example, instrumentation code can be dynamically inserted into Application Program **18** at runtime.

In other embodiments, Application Program **18** can be manually instrumented. In one example, a programmer can instrument a function call by placing an instrumenting instruction immediately after the function call as in the following example.

```
Device1.moveForward(12);
traceApplication(Device1.moveForward(12);');
```

In another example, an instrumenting instruction can be placed immediately before the function call, or at the beginning, end, or anywhere within the function itself. A programmer may instrument all function calls or only function calls of interest. In a further example, a programmer can instrument all lines of code or only code lines of interest. In a further example, a programmer can instrument other elements utilized or implemented within Application Program **18** such as objects and/or any of their functions, data structures and/or any of their functions, event handlers and/or any of their functions, threads and/or any of their functions, and/or other elements or functions. Similar instrumentation as in the preceding examples can be performed automatically or dynamically. In some designs where manual code instrumentation is utilized, Acquisition Interface **120** can optionally be omitted and Application Program's **18** instruction sets, data, and/or other information may be transmitted directly to Artificial Intelligence Unit **110**.

In some embodiments, DCADO Unit **100** can be selective in learning instruction sets, data, and/or other information to those implemented, utilized, or related to an object, data structure, repository, thread, function, and/or other element of Application Program **18**. In some aspects, Acquisition Interface **120** can obtain Application Program's **18** instruction sets, data, and/or other information implemented, utilized, or related to a certain object in an object oriented Application Program **18**.

In some embodiments, various computing systems and/or platforms may provide native tools for obtaining instruction sets, data, and/or other information. Also, independent vendors may provide portable tools with similar functionalities that can be utilized across different computing systems and/or platforms. These native and portable tools may provide a wide range of functionalities to obtain runtime and other information such as instrumentation, tracing or profiling, logging application or system messages, outputting custom text messages, outputting objects or data structures, outputting functions/routines/subroutines or their invocations, outputting variable or parameter values, outputting thread or process behaviors, outputting call or other stacks, outputting processor registers, providing runtime memory access, providing inputs and/or outputs, performing live application monitoring, and/or other capabilities. One of ordinary skill in art will understand that, while all possible variations of the techniques to obtain instruction sets, data, and/or other information are too voluminous to describe, these techniques are within the scope of this disclosure.

In one example, obtaining instruction sets, data, and/or other information can be implemented through the .NET platform's native tools for application tracing or profiling such as System.Diagnostics.Trace, System.Diagnostics.Debug, and System.Diagnostics.TraceSource classes for tracing execution flow, and System. Diagnostics. Process, System.Diagnostics.EventLog, and System. Diagnostics. PerformanceCounter classes for profiling code, accessing local and remote processes, starting and stopping system processes, and interacting with Windows event logs, etc. For instance, a set of trace switches can be created that output an application's information. The switches can be configured using the .config file. For a Web application, this may typically be Web.config file associated with the project. In a Windows application, this file may typically be named application Name.exe.config. Trace code can be added to application code automatically or manually as previously described. Appropriate listener can be created where the trace output is received. Trace code may output trace messages to a specific target such as a file, a log, a database, an object, a data structure, and/or other repository or system. Acquisition Interface **120** or Artificial Intelligence Unit **110** can then read or obtain the trace information from these targets. In some aspects, trace code may output trace messages directly to Acquisition Interface **120**. In other aspects, trace code may output trace messages directly to Artificial Intelligence Unit **110**. In the case of outputting trace messages to Acquisition Interface **120** or directly to Artificial Intelligence Unit **110**, custom listeners can be built to accommodate these specific targets. Other platforms, tools, and/or techniques can provide equivalent or similar functionalities as the above described ones.

In another example, obtaining instruction sets, data, and/or other information can be implemented through the .NET platform's Profiling API that can be used to create a custom profiler application for tracing, monitoring, interfacing with, and/or managing a profiled application. The Profiling API provides an interface that includes methods to notify the

profiler of events in the profiled application. The Profiling API may also provide an interface to enable the profiler to call back into the profiled application to obtain information about the state of the profiled application. The Profiling API may further provide call stack profiling functionalities. Call stack (also referred to as execution stack, control stack, runtime stack, machine stack, the stack, etc.) includes a data structure that can store information about active subroutines of an application. The Profiling API may provide a stack snapshot method, which enables a trace of the stack at a particular point in time. The Profiling API may also provide a shadow stack method, which tracks the call stack at every instant. A shadow stack can obtain function arguments, return values, and information about generic instantiations. A function such as FunctionEnter can be utilized to notify the profiler that control is being passed to a function and can provide information about the stack frame and function arguments. A function such as FunctionLeave can be utilized to notify the profiler that a function is about to return to the caller and can provide information about the stack frame and function return value. An alternative to call stack profiling includes call stack sampling in which the profiler can periodically examine the stack. In some aspects, the Profiling API enables the profiler to change the in-memory code stream for a routine before it is just-in-time (JIT) compiled where the profiler can dynamically add instrumentation code to all or particular routines of interest. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through Java platform's APIs for application tracing or profiling such as Java Virtual Machine Profiling Interface (JVMPI), Java Virtual Machine Tool Interface (JVMTI), and/or other APIs or tools. These APIs can be used for instrumentation of an application, for notification of Java Virtual Machine (VM) events, and/or other functionalities. One of the tracing or profiling techniques that can be utilized includes bytecode instrumentation. The profiler can insert bytecodes into all or some of the classes. In application execution profiling, for example, these bytecodes may include methodEntry and methodExit calls. In memory profiling, for example, the bytecodes may be inserted after each new or after each constructor. In some aspects, insertion of instrumentation bytecode can be performed either by a post-compiler or a custom class loader. An alternative to bytecode instrumentation includes monitoring events generated by the JVMPI or JVMTI interfaces. Both APIs can generate events for method entry/exit, object allocation, and/or other events. In some aspects, JVMTI can be utilized for dynamic bytecode instrumentation where insertion of instrumentation bytecodes is performed at runtime. The profiler may insert the necessary instrumentation when a selected class is invoked in an application. This can be accomplished using the JVMTI's redefineClasses method, for example. This approach also enables changing of the level of profiling as the application is running. If needed, these changes can be made adaptively without restarting the application. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through JVMTI's programming interface that enables creation of software agents that can monitor and control a Java application. An agent may use the functionality of the interface to register for notification of events as they occur in the application, and to query and control the application. A JVMTI agent

may use JVMTI functions to extract information from a Java application. A JVMTI agent can be utilized to obtain an application's runtime information such as method calls, memory allocation, CPU utilization, lock contention, and/or other information. JVMTI may include functions to obtain information about variables, fields, methods, classes, and/or other information. JVMTI may also provide notification for numerous events such as method entry and exit, exception, field access and modification, thread start and end, and/or other events. Examples of JVMTI built-in methods include GetMethodName to obtain the name of an invoked method, GetThreadInfo to obtain information for a specific thread, GetClassSignature to obtain information about the class of an object, GetStackTrace to obtain information about the stack including information about stack frames, and/or other methods. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through java.lang.Runtime class that provides an interface for application tracing or profiling. Examples of methods provided in java.lang.Runtime that can be used to obtain an application's instruction sets, data, and/or other information include tracemethodcalls, traceinstructions, and/or other methods. These methods prompt the Java Virtual Machine to output trace information for a method or instruction in the virtual machine as it is executed. The destination of trace output may be system dependent and include a file, a listener, and/or other destinations where Acquisition Interface **120**, Artificial Intelligence Unit **110**, and/or other disclosed elements can access needed information. In addition to tracing or profiling tools native to their respective computing systems and/or platforms, many independent tools exist that provide tracing or profiling functionalities on more than one computing system and/or platform. Examples of these tools include Pin, DynamoRIO, KernInst, DynInst, Kprobes, OpenPAT, DTrace, SystemTap, and/or others. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through logging tools of the platform and/or operating system on which an application runs. Some logging tools may include nearly full feature sets of the tracing or profiling tools previously described. In one example, Visual Basic enables logging of runtime messages through its Microsoft.VisualBasic.Logging namespace that provides a log listener where the log listener may direct logging output to a file and/or other target. In another example, Java enables logging through its java.util.logging class. In some aspects, obtaining an application's instruction sets, data, and/or other information can be implemented through logging capabilities of the operating system on which an application runs. For example, Windows NT features centralized log service that applications and operating-system components can utilize to report their events including any messages. Windows NT provides functionalities for system, application, security, and/or other logging. An application log may include events logged by applications. Windows NT, for example, may include support for defining an event source (i.e. application that created the event, etc.). Windows Vista, for example, supports a structured XML log-format and designated log types to allow applications to more precisely log events and to help interpret the events. Examples of different types of event logs include administrative, operational, analytic, debug, and/or other log types including any of their subcategories. Examples of event attributes that can be utilized include eventID, level, task, opcode, keywords, and/or other event attributes. Windows wevtutil tool enables access to events, their structures, registered event publishers, and/or their configuration even before the events are fired. Wevtutil supports capabilities such as retrieval of the names of all logs on a computing device; retrieval of configuration information for a specific log; retrieval of event publishers on a computing device; reading events from an event log, from a log file, or using a structured query; exporting events from an event log, from a log file, or using a structured query to a specific target; and/or other capabilities. Operating system logs can be utilized solely if they contain sufficient information on an application's instruction sets, data, and/or other information. Alternatively, operating system logs can be utilized in combination with another source of information (i.e. trace information, call stack, processor registers, memory, etc.) to reconstruct the application's instruction sets, data, and/or other information needed for Artificial Intelligence Unit **110** and/or other elements. In addition to logging capabilities native to their respective platforms and/or operating systems, many independent tools exist that provide logging on different platforms and/or operating systems. Examples of these tools include Log 4j, Logback, SmartInspect, NLog, log 4net, Microsoft Enterprise Library, ObjectGuy Framework, and/or others. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through tracing or profiling the operating system on which an application runs. As in tracing or profiling an application, one of the techniques that can be utilized includes adding instrumentation code to the operating system's source code. Such instrumentation code can be added to the operating system's source code before kernel compilation or recompilation, for instance. This type of instrumentation may involve defining or finding locations in the operating system's source code where instrumentation code may be inserted. Kernel instrumentation can also be performed without the need for kernel recompilation or rebooting. In some aspects, instrumentation code can be added at locations of interest through binary rewriting of compiled kernel code. In other aspects, kernel instrumentation can be performed dynamically where instrumentation code is added and/or removed where needed at runtime. For instance, dynamic instrumentation may overwrite kernel code with a branch instruction that redirects execution to instrumentation code or instrumentation routine. In yet other aspects, kernel instrumentation can be performed using just-in-time (JIT) dynamic instrumentation where execution may be redirected to a copy of kernel's code segment that includes instrumentation code. This type of instrumentation may include a JIT compiler and creation of a copy of the original code segment having instrumentation code or calls to instrumentation routines embedded into the original code segment. Instrumentation of the operating system may enable total system visibility including visibility into an application's behavior by enabling generation of low level trace information. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through tracing or profiling the processor on which an application runs. For example, some Intel processors provide Intel Processor Trace (i.e. Intel PT, etc.), a low-level tracing feature that enables recording executed instruction sets, and/or other

data or information of one or more applications. Intel PT is facilitated by the Processor Trace Decoder Library along with its related tools. Intel PT is a low-overhead execution tracing feature that records information about application execution on each hardware thread using dedicated hardware facilities. The recorded execution/trace information is collected in data packets that can be buffered internally before being sent to a memory subsystem or another system or element (i.e. Acquisition Interface **120**, Artificial Intelligence Unit **110**, etc.). Intel PT also enables navigating the recorded execution/trace information via reverse stepping commands. Intel PT can be included in an operating system's core files and provided as a feature of the operating system. Intel PT can trace globally some or all applications running on an operating system. Acquisition Interface **120** or Artificial Intelligence Unit **110** can read or obtain the recorded execution/trace information from Intel PT. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through branch tracing or profiling. Branch tracing may include an abbreviated instruction trace in which only the successful branch instruction sets are traced or recorded. Branch tracing can be implemented through utilizing dedicated processor commands, for example. Executed branches may be saved into special branch trace store area of memory. With the availability and reference to a compiler listing of the application together with branch trace information, a full path of executed instruction sets can be reconstructed. The full path can also be reconstructed with a memory dump (containing the program storage) and branch trace information. In some aspects, branch tracing can be utilized for pre-learning or automated learning of an application's instruction sets, data, and/or other information where a number of application simulations (i.e. simulations of likely/common operations, etc.) are performed. As such, the application's operation can be learned automatically saving the time that would be needed to learn the application's operation directed by a user. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through assembly language. Assembly language is a low-level programming language for a computer or other programmable device in which there is a strong correlation between the language and the architecture's machine instruction sets. Syntax, addressing modes, operands, and/or other elements of an assembly language instruction set may translate directly into numeric (i.e. binary, etc.) representations of that particular instruction set. Because of this direct relationship with the architecture's machine instruction sets, assembly language can be a powerful tool for tracing or profiling an application's execution in processor registers, memory, and/or other computing system components. For example, using assembly language, memory locations of a loaded application can be accessed, instrumented, and/or otherwise manipulated. In some aspects, assembly language can be used to rewrite or overwrite original in-memory instruction sets of an application with instrumentation instruction sets. In other aspects, assembly language can be used to redirect application's execution to instrumentation routine/subroutine or other code segment elsewhere in memory by inserting a jump into the application's in-memory code, by redirecting program counter, or by other techniques. Some operating systems may implement protection from changes

to applications loaded into memory. Operating system, processor, or other low level commands such as Linux mprotect command or similar commands in other operating systems may be used to unprotect the protected locations in memory before the change. In yet other aspects, assembly language can be used to obtain instruction sets, data, and/or other information through accessing and/or reading instruction register, program counter, other processor registers, memory locations, and/or other components of a computing system. In yet other aspects, high-level programming languages may call or execute an external assembly language program to facilitate obtaining instruction sets, data, and/or other information as previously described. In yet other aspects, relatively low-level programming languages such as C may allow embedding assembly language directly in their source code such as, for example, using asm keyword of C. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, it may be sufficient to obtain user or other inputs, variables, parameters, and/or other data in some procedural, simple object oriented, or other applications. In one instance, a simple procedural application executes a sequence of instruction sets until the end of the program. During its execution, the application may receive user or other input, store the input in a variable, and perform calculations using the variable to reach a result. The value of the variable can be obtained or traced. In another instance, a more complex procedural application comprises one or more functions/routines/subroutines each of which may include a sequence of instruction sets. The application may execute a main sequence of instruction sets with a branch to a function/routine/subroutine. During its execution, the application may receive user or other input, store the input in a variable, and pass the variable as a parameter to the function/routine/subroutine. The function/routine/subroutine may perform calculations using the parameter and return a value that the rest of the application can use to reach a result. The value of the variable or parameter passed to the function/routine/subroutine, and/or return value can be obtained or traced. Values of user or other inputs, variables, parameters, and/or other items of interest can be obtained through previously described tracing, instrumentation, and/or other techniques. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

Referring to FIG. **5**, in yet another example, obtaining instruction sets, data, and/or other information may be implemented through tracing, profiling, or sampling of instruction sets or data in processor registers, memory, or other computing system components where instruction sets, data, and/or other information may be stored or utilized. For example, Instruction Register **212** may be part of Processor **11** and it may store the instruction set currently being executed or decoded. In some processors, Program Counter **211** (also referred to as instruction pointer, instruction address register, instruction counter, or part of instruction sequencer) may be incremented after fetching an instruction set, and it may hold or point to the memory address of the next instruction set to be executed. In a processor where the incrementation precedes the fetch, Program Counter **211** may point to the current instruction set being executed. In the instruction cycle, an instruction set may be loaded into Instruction Register **212** after Processor **11** fetches it from location in Memory **12** pointed to by Program Counter **211**. Instruction Register **212** may hold the instruction set while it is decoded by Instruction Decoder **213**, prepared, and executed. In some aspects, data (i.e. operands, etc.) needed

for instruction set execution may be loaded from Memory **12** into a register within Register Array **214**. In other aspects, the data may be loaded directly into Arithmetic Logic Unit **215**. For instance, as instruction sets pass through Instruction Register **212** during application execution, they may be transmitted to Acquisition Interface **120** as shown. Examples of the steps in execution of a machine instruction set may include decoding the opcode (i.e. portion of a machine instruction set that may specify the operation to be performed), determining where the operands may be located (depending on architecture, operands may be in registers, the stack, memory, I/O ports, etc.), retrieving the operands, allocating processor resources to execute the instruction set (needed in some types of processors), performing the operation indicated by the instruction set, saving the results of execution, and/or other execution steps. Examples of the types of machine instruction sets that can be utilized include arithmetic, data handling, logical, program control, as well as special and/or other instruction set types. In addition to the ones described or shown, examples of other computing system or processor components that can be used during an instruction cycle include memory address register (MAR) that may hold the address of a memory block to be read from or written to; memory data register (MDR) that may hold data fetched from memory or data waiting to be stored in memory; data registers that may hold numeric values, characters, small bit arrays, or other data; address registers that may hold addresses used by instruction sets that indirectly access memory; general purpose registers (GPRs) that may store both data and addresses; conditional registers that may hold truth values often used to determine whether some instruction set should or should not be executed; floating point registers (FPRs) that may store floating point numbers; constant registers that may hold read-only values such as zero, one, or pi; special purpose registers (SPRs) such as status register, program counter, or stack pointer that may hold information on program state; machine-specific registers that may store data and settings related to a particular processor; Register Array **214** that may include an array of any number of processor registers; Arithmetic Logic Unit **215** that may perform arithmetic and logic operations; control unit that may direct processor's operation; and/or other circuits or components. Tracing, profiling, or sampling of processor registers, memory, or other computing system components can be implemented in a program, combination of hardware and program, or purely hardware system. Dedicated hardware may be built to perform tracing, profiling, or sampling of processor registers or any computing system components with marginal or no impact to computing overhead.

One of ordinary skill in art will recognize that FIG. **5** depicts one of many implementations of processor or computing system components, and that various additional components can be included, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate implementations. Processor or computing system components may be arranged or connected differently in alternate implementations. Processor or computing system components may also be connected with external elements using various connections. For instance, the connection between Instruction Register **212** and Acquisition Interface **120** may include any number or types of connections such as, for example, a dedicated connection for each bit of Instruction Register **212** (i.e. **32** connections for a 32 bit Instruction Register **212**, etc.). Any of the described or other connections or interfaces may be implemented among any processor or computing system components and Acquisition Interface **120** or other elements.

Referring to FIGS. **6**A-**6**B, in yet another example, obtaining instruction sets, data, and/or other information may be implemented through tracing, profiling, or sampling of Logic Circuit **250**. While Processor **11** includes any type or embodiment of logic circuit, Logic Circuit **250** is described separately here to offer additional detail on its functioning. Some Devices **98** may not need the processing capabilities of an entire Processor **11**, but instead a more tailored Logic Circuit **250**. Examples of such Devices **98** include home appliances, audio or video electronics, vehicle systems, toys, industrial machines, robots, and/or others. Logic Circuit **250** comprises the functionality for performing logic operations. Logic Circuit **250** comprises the functionality for performing logic operations using the circuit's inputs and producing outputs based on the logic operations performed on the inputs. Logic Circuit **250** may generally be implemented using transistors, diodes, and/or other electronic switches, but can also be constructed using vacuum tubes, electromagnetic relays (relay logic), fluidic logic, pneumatic logic, optics, molecules, or even mechanical elements. In some aspects, Logic Circuit **250** may be or include a microcontroller, field-programmable gate array (FPGA), application-specific integrated circuit (ASIC), and/or other computing circuit or device. In other aspects, Logic Circuit **250** may be or include any circuit or device comprising one or more logic gates, one or more transistors, one or more switches, and/or one or more other logic components. In further aspects, Logic Circuit **250** may be or include any integrated or other circuit or device that can perform logic operations. Logic may generally refer to Boolean logic utilized in binary operations, but other logics can also be used. Input into Logic Circuit **250** may include or refer to a value inputted into the Logic Circuit **250**, therefore, these terms may be used interchangeably herein depending on context. In one example, Logic Circuit **250** may perform some logic operations using four input values and produce two output values. As the four input values are delivered to or received by Logic Circuit **250**, they may be obtained by Acquisition Interface **120** through the four hardwired connections as shown in FIG. **6**A. In another example, Logic Circuit **250** may perform some logic operations using four input values and produce two output values. As the two output values are generated by or transmitted out of Logic Circuit **250**, they may be obtained by Acquisition Interface **120** through the two hardwired connections as shown in FIG. **6**B. In a further example, instead of or in addition to obtaining input and/or output values of Logic Circuit **250**, the state of Logic Circuit **250** may be obtained by reading or accessing values from one or more Logic Circuit's **250** internal components such as registers, memories, buses, and/or others (i.e. similar to the previously described tracing, profiling, and/or sampling of Processor **11** components, etc.). Tracing, profiling, or sampling of Logic Circuit **250** can be implemented in a program, combination of hardware and program, or purely hardware system. Dedicated hardware may be built to perform tracing, profiling, or sampling of Logic Circuit **250** with marginal or no impact to computing overhead. Any of the elements and/or techniques for tracing, profiling, or sampling of Logic Circuit **250** can similarly be implemented with Processor **11** and/or other processing elements. In some designs, DCADO Unit **100** may include clamps and/or other elements to attach DCADO Unit **100** to inputs (i.e. input wires, etc.) into and/or outputs (i.e. output wires, etc.) from Logic Circuit **250**. Such clamps and/or attachment elements enable seamless attachment of

DCADO Unit **100** to any circuit or computing device without the need to redesign or alter the circuit or computing device.

In some embodiments, DCADO Unit **100** may learn input values directly from an actuator (previously described, not shown). For example, Logic Circuit **250** or other processing element may control an actuator that enables Device **98** to perform mechanical, physical, and/or other operations. An actuator may receive one or more input values or control signals from Logic Circuit **250** or other processing element directing the actuator to perform specific operations. As one or more input values or control signals are delivered to or received by the actuator, they may be obtained by Acquisition Interface **120** as previously described with respect to obtaining input values of Logic Circuit **250**. Specifically, for instance, one or more input values or control signals of an actuator may be obtained by Acquisition Interface **120** via hardwired or other connections.

One of ordinary skill in art will recognize that FIGS. **6**A-**6**B depict one of many implementations of Logic Circuit **250** and that any number of input and/or output values can be utilized in alternate implementations. One of ordinary skill in art will also recognize that Logic Circuit **250** may include any number and/or combination of logic components to implement any logic operations.

Other additional techniques or elements may be utilized as needed for obtaining instruction sets, data, and/or other information, or some of the disclosed techniques or elements may be excluded, or a combination thereof may be utilized in alternate embodiments.

Referring to FIGS. **7**A-**7**E, some embodiments of Instruction Sets **526** are illustrated. In some aspects, Instruction Set **526** includes one or more instructions or commands of Application Program **18**. For example, Instruction Set **526** may include one or more instructions or commands of a high-level programming language such as Java or SQL, a low-level language such as assembly or machine language, an intermediate language or construct such as bytecode, and/or any other language or construct. In other aspects, Instruction Set **526** includes one or more inputs into and/or outputs from Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element. In further aspects, Instruction Set **526** includes one or more values or states of registers and/or other components of Logic Circuit **250**, Processor **11**, and/or other processing element. In general, Instruction Set **526** may include one or more instructions, commands, keywords, symbols (i.e. parentheses, brackets, commas, semicolons, etc.), operators (i.e. =, <, >, etc.), variables, values, objects, data structures, functions (i.e. Function1( ), FIRST( ), MIN( ), SQRT( ), etc.), parameters, states, signals, inputs, outputs, characters, digits, references thereto, and/or other components for performing an operation.

In an embodiment shown in FIG. **7**A, Instruction Set **526** includes code of a high-level programming language (i.e. Java, C++, etc.) comprising the following function call construct: Function1 (Parameter1, Parameter2, Parameter3, . . .). An example of a function call applying the above construct includes the following Instruction Set **526**: moveTo(Device1, 14, 8). The function or reference thereto "moveTo(Device1, 14, 8)" may be an Instruction Set **526** directing Device1 to move to a location with coordinates 14 and 8, for example. In another embodiment shown in FIG. **7**B, Instruction Set **526** includes structured query language (SQL). In a further embodiment shown in FIG. **7**C, Instruction Set **526** includes bytecode (i.e. Java bytecode, Python bytecode, CLR bytecode, etc.). In a further embodiment shown in FIG. **7**D, Instruction Set **526** includes assembly code. In a further embodiment shown in FIG. **7**E, Instruction Set **526** includes machine code.

Referring to FIGS. **8**A-**8**B, some embodiments of Extra Information **527** (also referred to as Extra Info **527**) are illustrated. In an embodiment shown in FIG. **8**A, Collection of Object Representations **525** may include or be associated with Extra Info **527**. In an embodiment shown in FIG. **8**B, Instruction Set **526** may include or be associated with Extra Info **527**.

Extra Info **527** comprises the functionality for storing any information useful in comparisons or decision making performed in autonomous device operation, and/or other functionalities. One or more Extra Infos **527** can be stored in, appended to, or associated with a Collection of Object Representations **525**, Instruction Set **526**, and/or other element. In some embodiments, the system can obtain Extra Info **527** at a time of creating or generating Collection of Object Representations **525**. In other embodiments, the system can obtain Extra Info **527** at a time of acquiring Instruction Set **526**. In general, Extra Info **527** can be obtained at any time. Examples of Extra Info **527** include time information, location information, computed information, contextual information, and/or other information. Any information can be utilized that can provide information for enhanced comparisons or decision making performed in autonomous device operation. Which information is utilized and/or stored in Extra Info **527** can be set by a user, by DCADO system administrator, or automatically by the system. Extra Info **527** may include or be referred to as contextual information, and vice versa. Therefore, these terms may be used interchangeably herein depending on context.

In some aspects, time information (i.e. time stamp, etc.) can be utilized and/or stored in Extra Info **527**. Time information can be useful in comparisons or decision making performed in autonomous device operation related to a specific time period as Device **98** may be required to perform specific operations at certain parts of day, month, year, and/or other time periods. Time information can be obtained from the system clock, online clock, oscillator, or other time source. In general, Extra Info **527** may include time information related to when Device **98** performed an operation. In other aspects, location information (i.e. coordinates, distance/angle from a known point, address, etc.) can be utilized and/or stored in Extra Info **527**. Location information can be useful in comparisons or decision making performed in autonomous device operation related to a specific place as Device **98** may be required to perform specific operations at certain places. Location information can be obtained from a positioning system (i.e. radio signal triangulation, GPS capabilities, etc.), sensors, and/or other location system. In general, Extra Info **527** may include location information related to where Device **98** performed an operation. In further aspects, computed information can be utilized and/or stored in Extra Info **527**. Computed information can be useful in comparisons or decision making performed in autonomous device operation where information can be calculated, inferred, or derived from other available information. DCADO Unit **100** and/or other disclosed elements may include computational functionalities to create Extra Info **527** by performing calculations or inferences using other information. In one example, Device's **98** speed can be computed or estimated from Device's **98** location and/or time information. In another example, Device's **98** bearing (i.e. angle or direction of movement, etc.) can be computed or estimated from

Device's **98** location information by utilizing Pythagorean theorem, trigonometry, and/or other theorems, formulas, or disciplines. In a further example, speeds, bearings, distances, and/or other properties of objects around Device **98** can similarly be computed or inferred using known information. In further aspects, observed information can be utilized and/or stored in Extra Info **527**. In further aspects, other information can be utilized and/or stored in Extra Info **527**. Examples of such other information include user specific information (i.e. skill level, age, gender, etc.), group user information (i.e. access level, etc.), version of Application Program **18**, type of Application Program **18**, type of Processor **11**, type of Logic Circuit **250**, type of Device **98**, and/or other information.

Referring to FIG. **9**, an embodiment where DCADO Unit **100** is part of or operating on Processor **11** is illustrated. In one example, DCADO Unit **100** may be a hardware element or circuit embedded or built into Processor **11**. In another example, DCADO Unit **100** may be a program operating on Processor **11**.

Referring to FIG. **10**, an embodiment where DCADO Unit **100** resides on Server **96** accessible over Network **95** is illustrated. Any number of Devices **98** may connect to such remote DCADO Unit **100** and the remote DCADO

Unit **100** may learn their operations in circumstances including objects with various properties. In turn, any number of Devices **98** can utilize the remote DCADO Unit **100** for autonomous operation in circumstances including objects with various properties. A remote DCADO Unit **100** can be offered as a network service (i.e. online application, etc.). In some aspects, a remote DCADO Unit **100** (i.e. global DCADO Unit **100**, etc.) may reside on the Internet and be available to all the world's Devices **98** configured to transmit their operations in circumstances including objects with various properties and/or configured to utilize the remote DCADO Unit **100** for autonomous operation in circumstances including objects with various properties. For example, multiple operators (i.e. Users **50**, etc.) may operate their Devices **98** where the Devices **98** may be configured to transmit their operations in circumstances including objects with various properties to a remote DCADO Unit **100**. Such remote DCADO Unit **100** enables learning of the operators' collective knowledge of operating Device **98** in circumstances including objects with various properties. Server **96** may be or include any type or form of a remote computing device such as an application server, a network service server, a cloud server, a cloud, and/or other remote computing device. Server **96** may include any features, functionalities, and embodiments of the previously described Computing Device **70**. It should be understood that Server **96** does not have to be a separate computing device and that Server **96**, its elements, or its functionalities can be implemented on Device **98**. Network **95** may include various networks, connection types, protocols, interfaces, APIs, and/or other elements or techniques known in art all of which are within the scope of this disclosure. Any of the previously described networks, network or connection types, networking interfaces, and/or other networking elements or techniques can similarly be utilized. Any of the disclosed elements may reside on Server **96** in alternate implementations. In one example, Artificial Intelligence Unit **110** can reside on Server **96** and Acquisition Interface **120** and/or Modification Interface **130** can reside on Device **98**. In another example, Knowledgebase **530** can reside on Server **96** and the rest of the elements of DCADO Unit **100** can reside on Device **98**. Any other combination of local and remote elements can be implemented.

Referring to FIG. **11**, an embodiment of learning and/or using Remote Device's **97** circumstances for autonomous Remote Device **97** operation is illustrated. In such embodiments, in addition to providing input into Object Processing Unit **93** for learning functionalities herein, Sensor **92** (i.e. Camera **92***a*, Radar **92***d*, Sonar **92***e*, etc.) can provide input into Display **21** or other device for User's **50** perception of Remote Device's **97** surrounding. As User **50** operates Remote Device **97**, DCADO Unit **100** may learn Remote Device's **97** operation in circumstances including objects with various properties. Such embodiments can be utilized in any situation where one device controls (i.e. remote controls, etc.) another device, any situation where some or all of the processing is on one device and sensor capabilities are on another device, and/or other situations. In one example, a drone controlling device (i.e. Device **98**, etc.) may send control signals to operate a drone (i.e. Remote Device **97**, etc.) and receive information on the drone's surrounding from Sensor **92** on the drone. In another example, a robot controlling device (i.e. Device **98**, etc.) may send control signals to operate a robot (i.e. Remote Device **97**, etc.) and receive information on the robot's surrounding from Sensor **92** on the robot. Any of the disclosed elements in addition to Sensor **92** may reside on Remote Device **97** in alternate implementations.

Referring to FIG. **12**, an embodiment of Artificial Intelligence Unit **110** is illustrated. Artificial Intelligence Unit **110** comprises interconnected Knowledge Structuring Unit **520**, Knowledgebase **530**, Decision-making Unit **540**, and Confirmation Unit **550**. Other additional elements can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate embodiments.

Artificial Intelligence Unit **110** comprises the functionality for learning Device's **98** operation in circumstances including objects with various properties. Artificial Intelligence Unit **110** comprises the functionality for learning one or more collections of object representations correlated with any instruction sets, data, and/or other information. In some aspects, Artificial Intelligence Unit **110** comprises the functionality for learning one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**. In other aspects, Artificial Intelligence Unit **110** comprises the functionality for learning one or more Collections of Object Representations **525** some of which may not be correlated with any Instruction Sets **526** and/or Extra Info **527**. Further, Artificial Intelligence Unit **110** comprises the functionality for anticipating Device's **98** operation in circumstances including objects with various properties. Artificial Intelligence Unit **110** comprises the functionality for anticipating one or more instruction sets, data, and/or other information. Artificial Intelligence Unit **110** comprises the functionality for anticipating one or more Instruction Sets **526** based on one or more incoming Collections of Object Representations **525**. Artificial Intelligence Unit **110** comprises the functionality for anticipating one or more Instruction Sets **526** to be used or executed in Device's **98** autonomous operation. Artificial Intelligence Unit **110** also comprises other disclosed functionalities.

Knowledge Structuring Unit **520**, Knowledgebase **530**, and Decision-making Unit **540** are described later.

Confirmation Unit **550** comprises the functionality for confirming, modifying, evaluating (i.e. rating, etc.), and/or canceling one or more anticipatory Instruction Sets **526**, and/or other functionalities. Confirmation Unit **550** is an optional element that can be omitted depending on implementation. In some embodiments, Confirmation Unit **550**

can serve as a means of confirming anticipatory Instruction Sets **526**. For example, Decision-making Unit **540** may determine one or more anticipatory Instruction Sets **526** and provide them to User **50** for confirmation. User **50** may be provided with an interface (i.e. graphical user interface, selectable list of anticipatory Instruction Sets **526**, etc.) to approve or confirm execution of the anticipatory Instruction Sets **526**. In some aspects, Confirmation Unit **550** can automate User **50** confirmation. In one example, if one or more incoming Collections of Object Representations **525** from Object Processing Unit **93** and one or more Collections of Object Representations **525** from a Knowledge Cell **800** were found to be a perfect or highly similar match, anticipatory Instruction Sets **526** correlated with the one or more Collections of Object Representations **525** from the Knowledge Cell **800** can be automatically executed without User's **50** confirmation. Conversely, if one or more incoming Collections of Object Representations **525** from Object Processing Unit **93** and one or more Collections of Object Representations **525** from a Knowledge Cell **800** were found to be less than a highly similar match, anticipatory Instruction Sets **526** correlated with the one or more Collections of Object Representations **525** from the Knowledge Cell **800** can be presented to User **50** for confirmation and/or modifying. Any features, functionalities, and/or embodiments of Similarity Comparison **125** (later described) can be utilized for such similarity determination. In other embodiments, Confirmation Unit **550** can serve as a means of modifying or editing anticipatory Instruction Sets **526**. For example, Decision-making Unit **540** may determine one or more anticipatory Instruction Sets **526** and provide them to User **50** for modification. User **50** may be provided with an interface (i.e. graphical user interface, etc.) to modify the anticipatory Instruction Sets **526** before their execution. In further embodiments, Confirmation Unit **550** can serve as a means of evaluating or rating anticipatory Instruction Sets **526** if they matched User's **50** intended operation. For example, Decision-making Unit **540** may determine one or more anticipatory Instruction Sets **526**, which the system may automatically execute. User **50** may be provided with an interface (i.e. graphical user interface, etc.) to rate (i.e. on a scale from 0 to 1, etc.) how well Decision-making Unit **540** predicted the executed anticipatory Instruction Sets **526**. In some aspects, rating can be automatic and based on a particular function or method that rates how well the anticipatory Instruction Sets **526** matched the desired operation. In one example, a rating function or method can assign a higher rating to anticipatory Instruction Sets **526** that were least modified in the confirmation process. In another example, a rating function or method can assign a higher rating to anticipatory Instruction Sets **526** that were canceled least number of times by User **50**. Any other automatic rating function or method can be utilized. In yet other embodiments, Confirmation Unit **550** can serve as a means of canceling anticipatory Instruction Sets **526** if they did not match User's **50** intended operation. For example, Decision-making Unit **540** may determine one or more anticipatory Instruction Sets **526**, which the system may automatically execute. The system may save the state of Device **98**, Processor **11** (save its register values, etc.), Logic Circuit **250**, Application Program **18** (i.e. save its variables, data structures, objects, location of its current instruction, etc.), and/or other processing elements before executing anticipatory Instruction Sets **526**. User **50** may be provided with an interface (i.e. graphical user interface, selectable list of prior executed anticipatory Instruction Sets **526**, etc.) to cancel one or more of the prior executed anticipatory Instruction Sets **526**, and restore Device **98**, Processor **11**, Logic Circuit **250**, Application Program **18**, and/or other processing elements to a prior state. In some aspects, Confirmation Unit **550** can optionally be disabled or omitted in order to provide an uninterrupted operation of Device **98**, Processor **11**, Logic Circuit **250**, and/or Application Program **18**. For example, a thermostat may be suitable for implementing the user confirmation step, whereas, a vehicle may be less suitable for implementing such interrupting step due to the real time nature of vehicle operation.

Referring to FIG. **13**, an embodiment of Knowledge Structuring Unit **520** correlating individual Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527** is illustrated. Knowledge Structuring Unit **520** comprises the functionality for structuring the knowledge of a device's operation in circumstances including objects with various properties, and/or other functionalities. Knowledge Structuring Unit **520** comprises the functionality for correlating one or more Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527**. Knowledge Structuring Unit **520** comprises the functionality for creating or generating Knowledge Cell **800** and storing one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** into the Knowledge Cell **800**. As such, Knowledge Cell **800** comprises the functionality for storing one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**. Knowledge Cell **800** includes knowledge (i.e. unit of knowledge, etc.) of how Device **98** operated in a circumstance including objects with various properties. Once created or generated, Knowledge Cells **800** can be used in/as neurons, nodes, vertices, or other elements in any of the data structures or arrangements (i.e. neural networks, graphs, sequences, etc.) used for storing the knowledge of Device's **98** operation in circumstances including objects with various properties, thereby facilitating learning functionalities herein. It should be noted that Extra Info **527** may be optionally used in some implementations to enable enhanced comparisons or decision making in autonomous device operation where applicable, and that Extra Info **527** can be omitted in alternate implementations.

In some embodiments, Knowledge Structuring Unit **520** receives one or more Collections of Object Representations **525** from Object Processing Unit **93**. Knowledge Structuring Unit **520** may also receive one or more Instruction Sets **526** from Acquisition Interface **120**. Knowledge Structuring Unit **520** may further receive any Extra Info **527**. Although, Extra Info **527** is not shown in this and/or other figures for clarity of illustration, it should be noted that any Collection of Object Representations **525**, Instruction Set **526**, and/or other element may include or be associated with Extra Info **527**. Knowledge Structuring Unit **520** may correlate one or more Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527**. Knowledge Structuring Unit **520** may then create Knowledge Cell **800** and store the one or more Collections of Object Representations **525** correlated with Instruction Sets **526** and/or Extra Info **527** into the Knowledge Cell **800**. Knowledge Cell **800** may include any data structure or arrangement that can facilitate such storing. For example, Knowledge Structuring Unit **520** may create Knowledge Cell **800*ax*** and structure within it Collection of Object Representations **525*a*1** correlated with Instruction Sets **526*a*1**-**526*a*3** and/or any Extra Info **527** (not shown). Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800*ax*** a Collection of Object Representations **525*a*2** correlated with Instruction

Set **526**_a_**4** and/or any Extra Info **527** (not shown). Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800**_ax_ a Collection of Object Representations **525**_a_**3** without a correlated Instruction Set **526** and/or Extra Info **527**. Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800**_ax_ a Collection of Object Representations **525**_a_**4** correlated with Instruction Sets **526**_a_**5**-**526**_a_**6** and/or any Extra Info **527** (not shown). Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800**_ax_ a Collection of Object Representations **525**_a_**5** without a correlated Instruction Set **526** and/or Extra Info **527**. Knowledge Structuring Unit **520** may structure within Knowledge Cell **800**_ax_ additional Collections of Object Representations **525** correlated with any number (including zero [i.e. uncorrelated]) of Instruction Sets **526** and/or Extra Info **527** by following similar logic as described above.

In some embodiments, Knowledge Structuring Unit **520** may correlate a Collection of Object Representations **525** with one or more temporally corresponding Instruction Sets **526** and/or Extra Info **527**. This way, Knowledge Structuring Unit **520** can structure the knowledge of Device's **98** operation at or around the time of generating Collections of Object Representations **525**. Such functionality enables spontaneous or seamless learning of Device's **98** operation in circumstances including objects with various properties as Device **98** is operated in real life situations. In some designs, Knowledge Structuring Unit **520** may receive a stream of Instruction Sets **526** used or executed to effect Device's **98** operations as well as a stream of Collections of Object Representations **525** as the operations are performed. Knowledge Structuring Unit **520** can then correlate Collections of Object Representations **525** from the stream of Collections of Object Representations **525** with temporally corresponding Instruction Sets **526** from the stream of Instruction Sets **526** and/or any Extra Info **527**. Collections of Object Representations **525** without a temporally corresponding Instruction Set **526** may be uncorrelated, for instance. In some aspects, Instruction Sets **526** and/or Extra Info **527** that temporally correspond to a Collection of Object Representations **525** may include Instruction Sets **526** used and/or Extra Info **527** obtained at the time of generating the Collection of Object Representations **525**. In other aspects, Instruction Sets **526** and/or Extra Info **527** that temporally correspond to a Collection of Object Representations **525** may include Instruction Sets **526** used and/or Extra Info **527** obtained within a certain time period before and/or after generating the Collection of Object Representations **525**. For example, Instruction Sets **526** and/or Extra Info **527** that temporally correspond to a Collection of Object Representations **525** may include Instruction Sets **526** used and/or Extra Info **527** obtained within 50 milliseconds, 1 second, 3 seconds, 20 seconds, 1 minute, 41 minutes, 2 hours, or any other time period before and/or after generating the Collection of Object Representations **525**. Such time periods can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. In other aspects, Instruction Sets **526** and/or Extra Info **527** that temporally correspond to a Collection of Object Representations **525** may include Instruction Sets **526** used and/or Extra Info **527** obtained from the time of generating the Collection of Object Representations **525** to the time of generating a next Collection of Object Representations **525**. In further aspects, Instruction Sets **526** and/or Extra Info **527** that temporally correspond to a Collection of Object Representations **525** may include Instruction Sets **526** used and/or Extra Info **527** obtained from the time of generating a previous Collection of Object Representations **525** to the time of generating the Collection of Object Representations **525**. Any other temporal relationship or correspondence between Collections of Object Representations **525** and correlated Instruction Sets **526** and/or Extra Info **527** can be implemented.

In some embodiments, Knowledge Structuring Unit **520** can structure the knowledge of Device's **98** operation in a circumstance including objects with various properties into any number of Knowledge Cells **800**. In some aspects, Knowledge Structuring Unit **520** can structure into a Knowledge Cell **800** a single Collection of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**. In other aspects, Knowledge Structuring Unit **520** can structure into a Knowledge Cell **800** any number (i.e. 2, 4, 7, 17, 29, 87, 1415, 23891, 323674, 8132401, etc.) of Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or

Extra Info **527**. In a special case, Knowledge Structuring Unit **520** can structure all Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** into a single long Knowledge Cell **800**. In further aspects, Knowledge Structuring Unit **520** can structure Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** into a plurality of Knowledge Cells **800**. In a special case, Knowledge Structuring Unit **520** can store periodic streams of Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** into a plurality of Knowledge Cells **800** such as hourly, daily, weekly, monthly, yearly, or other periodic Knowledge Cells **800**.

In some embodiments, Device **98** may include a plurality of Sensors **92** and/or their corresponding Object Processing Units **93**. In one example, multiple Sensors **92** may detect objects and/or their properties from different angles or on different sides of Device **98**. In another example, one or more Sensors **92** may be placed on different sub-devices, sub-systems, or elements of Device **98**. Using multiple Sensors **92** and/or their corresponding Object Processing Units **93** may provide additional detail in learning and/or using Device's **98** circumstances for autonomous Device **98** operation. In some designs where multiple Sensors **92** and/or their corresponding Object Processing Units **93** are utilized, multiple DCADO Units **100** can also be utilized (i.e. one DCADO Unit **100** for each Sensor **92** and its corresponding Object Processing Unit **93**, etc.). In such designs, Collections of Object Representations **525** can be correlated with any Instruction Sets **526** and/or Extra Info **527** as previously described. In other designs where multiple Sensors **92** and/or their corresponding Object Processing Units **93** are utilized, collective Collections of Object Representations **525** from multiple Sensors **92** and their corresponding Object Processing Units **93** can be correlated with any Instruction Sets **526** and/or Extra Info **527**.

In some embodiments, Device **98** may include a plurality of Logic Circuits **250**, Processors **11**, Application Programs **18**, and/or other processing elements. For example, each processing element may control a sub-device, sub-system, or an element of Device **98**. Using multiple processing elements may provide enhanced control over Device's **98** operation. In some designs where multiple processing elements are utilized, multiple DCADO Units **100** can also be utilized (i.e. one DCADO Unit **100** for each processing element, etc.). In such designs, Collections of Object Representations **525** can be correlated with any Instruction Sets

526 and/or Extra Info **527** as previously described. In other designs where multiple processing elements are utilized, Collections of Object Representations **525** can be correlated with any collective Instruction Sets **526** and/or Extra Info **527** used or executed by a plurality of processing elements.

Any combination of the aforementioned multiple Sensors **92** and/or their corresponding Object Processing Units **93**, multiple processing elements, and/or other elements can be implemented in alternate embodiments.

Referring to FIG. **14**, another embodiment of Knowledge Structuring Unit **520** correlating individual Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527** is illustrated. In such embodiments, Knowledge Structuring Unit **520** may generate Knowledge Cells **800** each comprising a single Collection of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**.

Referring to FIG. **15**, an embodiment of Knowledge Structuring Unit **520** correlating streams of Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527** is illustrated. For example, Knowledge Structuring Unit **520** may create Knowledge Cell **800***ax* and structure within it a stream of Collections of Object Representations **525***a***1**-**525***an* correlated with Instruction Set **526***a***1** and/or any Extra Info **527** (not shown). Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800***ax* a stream of Collections of Object Representations **525***b***1**-**525***bn* correlated with Instruction Sets **526***a***2**-**526***a***4** and/or and Extra Info **527** (not shown). Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800***ax* a stream of Collections of Object Representations **525***c***1**-**525***cn* without correlated Instruction Sets **526** and/or Extra Info **527**. Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800***ax* a stream of Collections of Object Representations **525***d***1**-**525***dn* correlated with Instruction Sets **526***a***5**-**526***a***6** and/or any Extra Info **527** (not shown). Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800***ax* additional streams of Collections of Object Representations **525** correlated with any number (including zero [i.e. uncorrelated]) of Instruction Sets **526** and/or Extra Info **527** by following similar logic as described above. The number of Collections of Object Representations **525** in some or all streams of Collections of Object Representations **525***a***1**-**525***an*, **525***b***1**-**525***bn*, etc. may be equal or different. It should be noted that n or other such letters or indicia may follow the sequence and/or context where they are indicated. Also, a same letter or indicia such as n may represent a different number in different elements of a drawing.

Referring to FIG. **16**, another embodiment of Knowledge Structuring Unit **520** correlating streams of Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527** is illustrated. In such embodiments, Knowledge Structuring Unit **520** may generate Knowledge Cells **800** each comprising a single stream of Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**.

Knowledgebase **530** comprises the functionality for storing the knowledge of a device's operation in circumstances including objects with various properties, and/or other functionalities. Knowledgebase **530** comprises the functionality for storing one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**. Knowledgebase **530** comprises the functionality for storing one or more Knowledge Cells **800** each including one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**. In some aspects, Collections of Object Representations **525** correlated with Instruction Sets **526** and/or Extra Info **527** can be stored directly within Knowledgebase **530** without using Knowledge Cells **800** as the intermediary data structures. In some embodiments, Knowledgebase **530** may be or include Neural Network **530***a* (later described). In other embodiments, Knowledgebase **530** may be or include Graph **530***b* (later described). In further embodiments, Knowledgebase **530** may be or include Collection of Sequences **530***c* (later described). In further embodiments, Knowledgebase **530** may be or include Sequence **533** (later described). In further embodiments, Knowledgebase **530** may be or include Collection of Knowledge Cells **530***d* (later described). In general, Knowledgebase **530** may be or include any data structure or arrangement capable of storing the knowledge of a device's operation in circumstances including objects with various properties. Knowledgebase **530** may reside locally on Device **98**, or remotely (i.e. remote Knowledgebase **530**, etc.) on a remote computing device (i.e. server, cloud, etc.) accessible over a network or an interface.

In some embodiments, Knowledgebase **530** from one Device **98** or DCADO Unit **100** can be transferred to one or more other Devices **98** or DCADO Units **100**. Therefore, the knowledge of Device's **98** operation in circumstances including objects with various properties learned on one Device **98** or DCADO Unit **100** can be transferred to one or more other Devices **98** or DCADO Units **100**. In one example, Knowledgebase **530** can be copied or downloaded to a file or other repository from one Device **98** or DCADO Unit **100** and loaded or inserted into another Device **98** or DCADO Unit **100**. In another example, Knowledgebase **530** from one Device **98** or DCADO Unit **100** can be available on a server accessible by other Devices **98** or DCADO Units **100** over a network or an interface. Once loaded into or accessed by a receiving Device **98** or DCADO Unit **100**, the receiving Device **98** or DCADO Unit **100** can then implement the knowledge of Device's **98** operation in circumstances including objects with various properties learned on the originating Device **98** or DCADO Unit **100**.

In some embodiments, multiple Knowledgebases **530** (i.e. Knowledgebases **530** from different Devices **98** or DCADO Units **100**, etc.) can be combined to accumulate collective knowledge of operating Device **98** in circumstances including objects with various properties. In one example, one Knowledgebase **530** can be appended to another Knowledgebase **530** such as appending one Collection of Sequences **530***c* (later described) to another Collection of Sequences **530***c*, appending one Sequence **533** (later described) to another Sequence **533**, appending one Collection of Knowledge Cells **530***d* (later described) to another Collection of Knowledge Cells **530***d*, and/or appending other data structures or elements thereof. In another example, one Knowledgebase **530** can be copied into another Knowledgebase **530** such as copying one Collection of Sequences **530***c* into another Collection of Sequences **530***c*, copying one Collection of Knowledge Cells **530***d* into another Collection of Knowledge Cells **530***d*, and/or copying other data structures or elements thereof. In a further example, in the case of Knowledgebase **530** being or including Graph **530***b* or graph-like data structure (i.e. Neural Network **530***a*, tree, etc.), a union can be utilized to combine two or more Graphs **530***b* or graph-like data structures. For instance, a union of two Graphs **530***b* or graph-like data structures may include a union of their vertex (i.e. node, etc.) sets and their edge (i.e. connection, etc.) sets. Any other

operations or combination thereof on graphs or graph-like data structures can be utilized to combine Graphs **530***b* or graph-like data structures. In a further example, one Knowledgebase **530** can be combined with another Knowledgebase **530** through later described learning processes where Knowledge Cells **800** may be applied one at a time and connected with prior and/or subsequent Knowledge Cells **800** such as in Graph **530***b* or Neural Network **530***a*. In such embodiments, instead of Knowledge Cells **800** generated by Knowledge Structuring Unit **520**, the learning process may utilize Knowledge Cells **800** from one Knowledgebase **530** to apply them onto another Knowledgebase **530**. Any other techniques known in art including custom techniques for combining data structures can be utilized for combining Knowledgebases **530** in alternate implementations. In any of the aforementioned and/or other combining techniques, similarity of elements (i.e. nodes/vertices, edges/connections, etc.) can be utilized in determining whether an element from one Knowledgebase **530** matches an element from another Knowledgebase **530**, and substantially or otherwise similar elements may be considered a match for combining purposes in some designs. Any features, functionalities, and embodiments of Similarity Comparison **125** (later described) can be used in such similarity determinations. A combined Knowledgebase **530** can be offered as a network service (i.e. online application, etc.), downloadable file, or other repository to all DCADO Units **100** configured to utilize the combined Knowledgebase **530**. For example, a Device **98** including or interfaced with DCADO Unit **100** having access to a combined Knowledgebase **530** can use the collective knowledge learned from multiple Devices **98** for the Device's **98** autonomous operation.

Referring to FIG. **17**, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation may include various artificial intelligence models and/or techniques. The disclosed systems, devices, and methods are independent of the artificial intelligence model and/or technique used and any model and/or technique can be utilized to facilitate the functionalities described herein. Examples of these models and/or techniques include deep learning, supervised learning, unsupervised learning, neural networks (i.e. convolutional neural network, recurrent neural network, deep neural network, etc.), search-based, logic and/or fuzzy logic-based, optimization-based, tree/graph/other data structure-based, hierarchical, symbolic and/or sub-symbolic, evolutionary, genetic, multi-agent, deterministic, probabilistic, statistical, and/or other models and/or techniques.

In one example shown in Model A, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation may include a neural network (also referred to as artificial neural network, etc.). As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include a network of Nodes **852** (also referred to as neurons, etc.) and Connections **853** similar to that of a brain. Node **852** can store any data, object, data structure, and/or other item, or reference thereto. Node **852** may also include a function for transforming or manipulating any data, object, data structure, and/or other item. Examples of such transformation functions include mathematical functions (i.e. addition, subtraction, multiplication, division, sin, cos, log, derivative, integral, etc.), object manipulation functions (i.e. creating an object, modifying an object, deleting an object, appending objects, etc.), data structure manipulation functions (i.e. creating a data structure, modifying a data structure, deleting a data structure, creating a data field, modifying a data field, deleting a data field, etc.), and/or other transformation functions. Connection **853** may include or be associated with a value such as a symbolic label or numeric attribute (i.e. weight, cost, capacity, length, etc.). A computational model can be utilized to compute values from inputs based on a pre-programmed or learned function or method. For example, a neural network may include one or more input neurons that can be activated by inputs. Activations of these neurons can then be passed on, weighted, and transformed by a function to other neurons. Neural networks may range from those with only one layer of single direction logic to multi-layer of multi-directional feedback loops. A neural network can use weights to change the parameters of the network's throughput. A neural network can learn by input from its environment or from self-teaching using written-in rules. A neural network can be utilized as a predictive modeling approach in machine learning. An exemplary embodiment of a neural network (i.e. Neural Network **530***a*, etc.) is described later.

In another example shown in Model B, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation may include a graph or graph-like data structure. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include Nodes **852** (also referred to as vertices or points, etc.) and Connections **853** (also referred to as edges, arrows, lines, arcs, etc.) organized as a graph. In general, any Node **852** in a graph can be connected to any other Node **852**. A Connection **853** may include unordered pair of Nodes **852** in an undirected graph or ordered pair of Nodes **852** in a directed graph. Nodes **852** can be part of the graph structure or external entities represented by indices or references. A graph can be utilized as a predictive modeling approach in machine learning. Nodes **852**, Connections **853**, and/or other elements or operations of a graph may include any features, functionalities, and embodiments of the aforementioned Nodes **852**, Connections **853**, and/or other elements or operations of a neural network, and vice versa. An exemplary embodiment of a graph (i.e. Graph **530***b*, etc.) is described later.

In a further example shown in Model C, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation may include a tree or tree-like data structure. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include Nodes **852** and Connections **853** (also referred to as references, edges, etc.) organized as a tree. In general, a Node **852** in a tree can be connected to any number (i.e. including zero, etc.) of children Nodes **852**. A tree can be utilized as a predictive modeling approach in machine learning. Nodes **852**, Connections **853**, and/or other elements or operations of a tree may include any features, functionalities, and embodiments of the aforementioned Nodes **852**, Connections **853**, and/or other elements or operations of a neural network and/or graph, and vice versa.

In a further example shown in Model D, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation may include a sequence or sequence-like data structure. As such, machine learning, knowledge structuring or representation, decision making, pattern rec-

ognition, and/or other artificial intelligence functionalities may include a structure of Nodes **852** and/or

Connections **853** organized as a sequence. In some aspects, Connections **853** may be optionally omitted from a sequence as the sequential order of Nodes **852** in a sequence may be implied in the structure. A sequence can be utilized as a predictive modeling approach in machine learning. Nodes **852**, Connections **853**, and/or other elements or operations of a sequence may include any features, functionalities, and embodiments of the aforementioned Nodes **852**, Connections **853**, and/or other elements or operations of a neural network, graph, and/or tree, and vice versa. An exemplary embodiment of a sequence (i.e. Collection of Sequences **530***c*, Sequence **533**, etc.) is described later.

In yet another example, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation may include a search-based model and/or technique. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include searching through a collection of possible solutions. For example, a search method can search through a neural network, graph, tree, sequence, or other data structure that includes data elements of interest. A search may use heuristics to limit the search for solutions by eliminating choices that are unlikely to lead to the goal. Heuristic techniques may provide a best guess solution. A search can also include optimization. For example, a search may begin with a guess and then refine the guess incrementally until no more refinements can be made. In a further example, the disclosed systems, devices, and methods may include logic-based model and/or technique. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities can use formal or other type of logic. Logic based models may involve making inferences or deriving conclusions from a set of premises. As such, a logic based system can extend existing knowledge or create new knowledge automatically using inferences. Examples of the types of logic that can be utilized include propositional or sentential logic that comprises logic of statements which can be true or false; first-order logic that allows the use of quantifiers and predicates and that can express facts about objects, their properties, and their relations with each other; fuzzy logic that allows degrees of truth to be represented as a value between 0 and 1 rather than simply 0 (false) or 1 (true), which can be used for uncertain reasoning; subjective logic that comprises a type of probabilistic logic that may take uncertainty and belief into account, which can be suitable for modeling and analyzing situations involving uncertainty, incomplete knowledge and different world views; and/or other types of logic. In a further example, the disclosed systems, devices, and methods may include a probabilistic model and/or technique. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities can be implemented to operate with incomplete or uncertain information where probabilities may affect outcomes. Bayesian network, among other models, is an example of a probabilistic tool used for purposes such as reasoning, learning, planning, perception, and/or others. One of ordinary skill in art will understand that the aforementioned artificial intelligence models and/or techniques are described merely as examples of a variety of possible implementations, and that while all possible artificial intelligence models and/or techniques are too voluminous to describe, other artificial intelligence models and/or techniques known in art are within the scope of this disclosure. One of ordinary skill in art will also recognize that an intelligent system may solve a specific problem by using any model and/or technique that works such as, for example, some systems can be symbolic and logical, some can be sub-symbolic neural networks, some can be deterministic or probabilistic, some can be hierarchical, some may include searching techniques, some may include optimization techniques, while others may use other or a combination of models and/or techniques. In general, any artificial intelligence model and/or technique can be utilized that can facilitate the functionalities described herein.

Referring to FIGS. **18**A-**18**C, embodiments of interconnected Knowledge Cells **800** and updating weights of Connections **853** are illustrated. As shown for example in FIG. **18**A, Knowledge Cell **800***za* is connected to Knowledge Cell **800***zb* and Knowledge Cell **800***zc* by Connection **853***z***1** and Connection **853***z***2**, respectively. Each of Connection **853***z***1** and Connection **853***z***2** may include or be associated with occurrence count, weight, and/or other parameter or data. The number of occurrences may track or store the number of observations that a Knowledge Cell **800** was followed by another Knowledge Cell **800** indicating a connection or relationship between them. For example, Knowledge Cell **800***za* was followed by Knowledge Cell **800***zb* 10 times as indicated by the number of occurrences of Connection **853***z***1**. Also, Knowledge Cell **800***za* was followed by Knowledge Cell **800***zc* 15 times as indicated by the number of occurrences of Connection **853***z***2**. The weight of Connection **853***z***1** can be calculated or determined as the number of occurrences of Connection **853***z***1** divided by the sum of occurrences of all connections (i.e. Connection **853***z***1** and Connection **853***z***2**, etc.) originating from Knowledge Cell **800***za*. Therefore, the weight of Connection **853***z***1** can be calculated or determined as 10/(10+15)=0.4, for example. Also, the weight of Connection **853***z***2** can be calculated or determined as 15/(10+15)=0.6, for example. Therefore, the sum of weights of Connection **853***z***1**, Connection **853***z***2**, and/or any other Connections **853** originating from Knowledge Cell **800***za* may equal to 1 or 100%. As shown for example in FIG. **18**B, in the case that Knowledge Cell **800***zd* is inserted and an observation is made that Knowledge Cell **800***zd* follows Knowledge Cell **800***za*, Connection **853***z***3** can be created between Knowledge Cell **800***za* and Knowledge Cell **800***zd*. The occurrence count of Connection **853***z***3** can be set to 1 and weight determined as 1/(10+15+1)=0.038. The weights of all other connections (i.e. Connection **853***z***1**, Connection **853***z***2**, etc.) originating from Knowledge Cell **800***za* may be updated to account for the creation of Connection **853***z***3**. Therefore, the weight of Connection **853***z***1** can be updated as 10/(10+15+1)=0.385. The weight of Connection **853***z***2** can also be updated as 15/(10+15+1)=0.577. As shown for example in FIG. **18**C, in the case that an additional occurrence of Connection **853***z***1** is observed (i.e. Knowledge Cell **800***zb* followed Knowledge Cell **800***za*, etc.), occurrence count of Connection **853***z***1** and weights of all connections (i.e. Connection **853***z***1**, Connection **853***z***2**, and Connection **853***z***3**, etc.) originating from Knowledge Cell **800***za* may be updated to account for this observation. The occurrence count of Connection **853***z***1** can be increased by 1 and its weight updated as 11/(11+15+1)=0.407. The weight of Connection **853***z***2** can also be updated as 15/(11+15+1)=0.556. The weight of Connection **853***z***3** can also be updated as 1/(11+15+1)=0.037.

Referring to FIG. **19**, an embodiment of learning Knowledge Cells **800** comprising one or more Collections of

Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Collection of Knowledge Cells **530***d* is illustrated. Collection of Knowledge Cells **530***d* comprises the functionality for storing any number of Knowledge Cells **800**. In some aspects, Knowledge Cells **800** may be stored into or applied onto Collection of Knowledge Cells **530***d* in a learning or training process. In effect, Collection of Knowledge Cells **530***d* may store Knowledge Cells **800** that can later be used to enable autonomous Device **98** operation. In some embodiments, Knowledge Structuring Unit **520** structures or generates Knowledge Cells **800** as previously described and the system applies them onto Collection of Knowledge Cells **530***d*, thereby implementing learning Device's **98** operation in circumstances including objects with various properties. The term apply or applying may refer to storing, copying, inserting, updating, or other similar action, therefore, these terms may be used interchangeably herein depending on context. The system can perform Similarity Comparisons **125** (later described) of a newly structured Knowledge Cell **800** from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. If a substantially similar Knowledge Cell **800** is not found in Collection of Knowledge Cells **530***d*, the system may insert (i.e. copy, store, etc.) the Knowledge Cell **800** from Knowledge Structuring Unit **520** into Collection of Knowledge Cells **530***d*, for example. On the other hand, if a substantially similar Knowledge Cell **800** is found in Collection of Knowledge Cells **530***d*, the system may optionally omit inserting the Knowledge Cell **800** from Knowledge Structuring Unit **520** as inserting a substantially similar Knowledge Cell **800** may not add much or any additional knowledge to the Collection of Knowledge Cells **530***d*, for example. Also, inserting a substantially similar Knowledge Cell **800** can optionally be omitted to save storage resources and limit the number of Knowledge Cells **800** that may later need to be processed or compared. Any features, functionalities, and embodiments of Similarity Comparison **125**, importance index (later described), similarity index (later described), and/or other disclosed elements can be utilized to facilitate determination of substantial or other similarity and whether to insert a newly structured Knowledge Cell **800** into Collection of Knowledge Cells **530***d*.

For example, the system can perform Similarity Comparisons **125** (later described) of Knowledge Cell **800***ba* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. In the case that a substantially similar match is found between Knowledge Cell **800***ba* and any of the Knowledge Cells **800** in Collection of Knowledge Cells **530***d*, the system may perform no action. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bb* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. In the case that a substantially similar match is not found, the system may insert a new Knowledge Cell **800** into Collection of Knowledge Cells **530***d* and copy Knowledge Cell **800***bb* into the inserted new Knowledge Cell **800**. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bc* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. In the case that a substantially similar match is found between Knowledge Cell **800***bc* and any of the Knowledge Cells **800** in Collection of Knowledge Cells **530***d*, the system may perform no action. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bd* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. In the case that a substantially similar match is not found, the system may insert a new Knowledge Cell **800** into Collection of Knowledge Cells **530***d* and copy Knowledge Cell **800***bd* into the inserted new Knowledge Cell **800**. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***be* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. In the case that a substantially similar match is not found, the system may insert a new Knowledge Cell **800** into Collection of Knowledge Cells **530***d* and copy Knowledge Cell **800***be* into the inserted new Knowledge Cell **800**. Applying any additional Knowledge Cells **800** from Knowledge Structuring Unit **520** onto Collection of Knowledge Cells **530***d* follows similar logic or process as the above-described.

Referring to FIG. **20**, an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Neural Network **530***a* is illustrated. Neural Network **530***a* includes a number of neurons or Nodes **852** interconnected by Connections **853** as previously described. Knowledge Cells **800** are shown instead of Nodes **852** to simplify the illustration as Node **852** includes a Knowledge Cell **800**, for example. Therefore, Knowledge Cells **800** and Nodes **852** can be used interchangeably herein depending on context. It should be noted that Node **852** may include other elements and/or functionalities instead of or in addition to Knowledge Cell **800**. In some aspects, Knowledge Cells **800** may be stored into or applied onto Neural Network **530***a* individually or collectively in a learning or training process. In some designs, Neural Network **530***a* comprises a number of Layers **854** each of which may include one or more Knowledge Cells **800**. Knowledge Cells **800** in successive Layers **854** can be connected by Connections **853**.

Connection **853** may include or be associated with occurrence count, weight, and/or other parameter or data as previously described. Neural Network **530***a* may include any number of Layers **854** comprising any number of Knowledge Cells **800**. In some aspects, Neural Network **530***a* may store Knowledge Cells **800** interconnected by Connections **853** where following a path through the Neural Network **530***a* can later be used to enable autonomous Device **98** operation. It should be understood that, in some embodiments, Knowledge Cells **800** in one Layer **854** of Neural Network **530***a* need not be connected only with Knowledge Cells **800** in a successive Layer **854**, but also in any other Layer **854**, thereby creating shortcuts (i.e. shortcut Connections **853**, etc.) through Neural Network **530***a*. A Knowledge Cell **800** can also be connected to itself such as, for example, in recurrent neural networks. In general, any Knowledge Cell **800** can be connected with any other Knowledge Cell **800** anywhere else in Neural Network **530***a*. In further embodiments, back-propagation of any data or information can be implemented. In one example, back-propagation of similarity (i.e. similarity index, etc.) of compared Knowledge Cells **800** in a path through Neural Network **530***a* can be implemented. In another example, back-propagation of errors can be implemented. Such back-propagations can then be used to adjust occurrence counts and/or weights of Connections **853** for better future predictions, for example. Any other back-propagation can be implemented for other purposes. Any combination of Nodes **852** (i.e. Nodes **852** comprising Knowledge Cells **800**, etc.), Connections **853**, Layers **854**, and/or other elements or techniques can be implemented in alternate embodiments. Neural Network **530***a* may include any type or form of a

neural network known in art such as a feed-forward neural network, a back-propagating neural network, a recurrent neural network, a convolutional neural network, deep neural network, and/or others including a custom neural network.

In some embodiments, Knowledge Structuring Unit **520** structures or generates Knowledge Cells **800** and the system applies them onto Neural Network **530***a*, thereby implementing learning Device's **98** operation in circumstances including objects with various properties. The system can perform Similarity Comparisons **125** (later described) of a Knowledge Cell **800** from Knowledge Structuring Unit **520** with Knowledge Cells **800** in a Layer **854** of Neural Network **530***a*. If a substantially similar Knowledge Cell **800** is not found in the Layer **854** of Neural Network **530***a*, the system may insert (i.e. copy, store, etc.) the Knowledge Cell **800** from Knowledge Structuring Unit **520** into the Layer **854** of Neural Network **530***a*, and create a Connection **853** to the inserted Knowledge Cell **800** from a Knowledge Cell **800** in a prior Layer **854** including assigning an occurrence count to the new Connection **853**, calculating a weight of the new Connection **853**, and updating any other Connections **853** originating from the Knowledge Cell **800** in the prior Layer **854**. On the other hand, if a substantially similar Knowledge Cell **800** is found in the Layer **854** of Neural Network **530***a*, the system may update occurrence count and weight of Connection **853** to that Knowledge Cell **800** from a Knowledge Cell **800** in a prior Layer **854**, and update any other Connections **853** originating from the Knowledge Cell **800** in the prior Layer **854**.

For example, the system can perform Similarity Comparisons **125** (later described) of Knowledge Cell **800***ba* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Layer **854***a* of Neural Network **530***a*. In the case that a substantially similar match is found between Knowledge Cell **800***ba* and Knowledge Cell **800***ea*, the system may perform no action since Knowledge Cell **800***ea* is the initial Knowledge Cell **800**. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bb* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Layer **854***b* of Neural Network **530***a*. In the case that a substantially similar match is found between Knowledge Cell **800***bb* and Knowledge Cell **800***eb*, the system may update occurrence count and weight of Connection **853***e***1** between Knowledge Cell **800***ea* and Knowledge Cell **800***eb*, and update weights of other Connections **853** originating from Knowledge Cell **800***ea* as previously described. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bc* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Layer **854***c* of Neural Network **530***a*. In the case that a substantially similar match is not found, the system may insert Knowledge Cell **800***ec* into Layer **854***c* and copy Knowledge Cell **800***bc* into the inserted Knowledge Cell **800***ec*. The system may also create Connection **853***e***2** between Knowledge Cell **800***eb* and Knowledge Cell **800***ec* with occurrence count of 1 and weight calculated based on the occurrence count as previously described. The system may also update weights of other Connections **853** (one in this example) originating from Knowledge Cell **800***eb* as previously described. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bd* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Layer **854***d* of Neural Network **530***a*. In the case that a substantially similar match is not found, the system may insert Knowledge Cell **800***ed* into Layer **854***d* and copy Knowledge Cell **800***bd* into the inserted Knowledge Cell **800***ed*. The system may also create Connection **853***e***3** between Knowledge Cell **800***ec* and Knowledge Cell **800***ed* with occurrence count of 1 and weight of 1. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***be* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Layer **854***e* of Neural Network **530***a*. In the case that a substantially similar match is not found, the system may insert Knowledge Cell **800***ee* into Layer **854***e* and copy Knowledge Cell **800***be* into the inserted Knowledge Cell **800***ee*. The system may also create Connection **853***e***4** between Knowledge Cell **800***ed* and Knowledge Cell **800***ee* with occurrence count of 1 and weight of 1. Applying any additional Knowledge Cells **800** from Knowledge Structuring Unit **520** onto Neural Network **530***a* follows similar logic or process as the above-described.

Referring now to Similarity Comparison **125**, Similarity Comparison **125** comprises the functionality for comparing or matching Knowledge Cells **800** or portions thereof, and/or other functionalities. Similarity Comparison **125** comprises the functionality for comparing or matching Collections of Object Representations **525** or portions thereof. Similarity Comparison **125** comprises the functionality for comparing or matching streams of Collections of Object Representations **525** or portions thereof. Similarity Comparison **125** comprises the functionality for comparing or matching Object Representations **625** or portions thereof. Similarity Comparison **125** comprises the functionality for comparing or matching Object Properties **630** or portions thereof. Similarity Comparison **125** comprises the functionality for comparing or matching Instruction Sets **526**, Extra Info **527**, text (i.e. characters, words, phrases, etc.), numbers, and/or other elements or portions thereof. Similarity Comparison **125** may include functions, rules, and/or logic for performing matching or comparisons and for determining that while a perfect match is not found, a partial or similar match has been found. In some aspects, a partial match may include a substantially or otherwise similar match, and vice versa. Therefore, these terms may be used interchangeably herein depending on context. As such, Similarity Comparison **125** may include determining substantial similarity or substantial match of compared elements. Although, substantial similarity or substantial match is frequently used herein, it should be understood that any level of similarity, however high or low, may be utilized as defined by the rules (i.e. thresholds, etc.) for similarity. The rules for similarity or similar match can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. In some designs, Similarity Comparison **125** comprises the functionality to automatically define appropriately strict rules for determining similarity of the compared elements. Similarity Comparison **125** can therefore set, reset, and/or adjust the strictness of the rules for finding or determining similarity of the compared elements, thereby fine tuning Similarity Comparison **125** so that the rules for determining similarity are appropriately strict. In some aspects, the rules for determining similarity may include a similarity threshold. As such, Similarity Comparison **125** can determine similarity of compared elements if their similarity exceeds a similarity threshold. In other aspects, the rules for determining similarity may include a difference threshold. As such, Similarity Comparison **125** can determine similarity of compared elements if their difference is lower than a difference threshold. In further aspects, the rules for determining similarity may include other thresholds. Similarity Comparison **125** enables comparing circumstances including objects with various properties and determining their similarity or match. In one

example, a circumstance including an object detected at a distance of 8 m and an angle/bearing of 64° relative to Device **98** may be found similar or matching by Similarity Comparison **125** to a circumstance including the same or similar object detected at a distance of 8.6 m and an angle/bearing of 59° relative to Device **98**. In another example, a circumstance including an object detected as a passenger vehicle may be found similar or matching by Similarity Comparison **125** to a circumstance including an object detected as a sport utility vehicle. In general, any one or more properties (i.e. existence, type, identity, distance, bearing/angle, location, shape/size, activity, etc.) of one or more objects can be utilized for determining similarity or match of circumstances including objects with various properties. Therefore, Similarity Comparison **125** provides flexibility in comparing and determining similarity of a variety of possible circumstances of Device **98**.

In some embodiments where compared Knowledge Cells **800** include a single Collection of Object Representations **525**, in determining similarity of Knowledge Cells **800**, Similarity Comparison **125** can perform comparison of individual Collections of Object Representations **525** or portions (i.e. Object Representations **625**, Object Properties **630**, etc.) thereof such as comparison of Collection of Object Representations **525** or portions thereof from one Knowledge Cell **800** with Collection of Object Representations **525** or portions thereof from another Knowledge Cell **800**. In some aspects, total equivalence is achieved when Collection of Object Representations **525** or portions thereof from one Knowledge Cell **800** matches Collection of Object Representations **525** or portions thereof from another Knowledge Cell **800**. If total equivalence is not found, Similarity Comparison **125** may attempt to determine substantial or other similarity of compared Knowledge Cells **800**.

In some embodiments, in determining substantial similarity of individually compared Collections of Object Representations **525** (i.e. Collections of Object Representations **525** from the compared Knowledge Cells **800**, etc.), Similarity Comparison **125** can compare one or more Object Representations **625** or portions (i.e. Object Properties **630**, etc.) thereof from one Collection of Object Representations **525** with one or more Object Representations **625** or portions thereof from another Collection of Object Representations **525**. In some aspects, total equivalence is found when all Object Representations **625** or portions thereof from one Collection of Object Representations **525** match all Object Representations **625** or portions thereof from another Collection of Object Representations **525**. In other aspects, if total equivalence is not found, Similarity Comparison **125** may attempt to determine substantial similarity of compared Collections of Object Representations **525**. In one example, substantial similarity can be achieved when most of the Object Representations **625** or portions thereof from the compared Collections of Object Representations **525** match or substantially match. In another example, substantial similarity can be achieved when at least a threshold number (i.e. 1, 2, 4, 7, 18, etc.) or percentage (i.e. 41%, 62%, 79%, 85%, 93%, etc.) of Object Representations **625** or portions thereof from the compared Collections of Object Representations **525** match or substantially match. Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching Object Representations **625** or portions thereof from the compared Collections of Object Representations **525** exceeds a threshold number (i.e. 1, 2, 4, 7, 18, etc.) or a threshold percentage (i.e. 41%, 62%, 79%, 85%, 93%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or percentage of Object Representations **625** or portions thereof from the compared Collections of Object Representations **525** match or substantially match. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In some aspects, Similarity Comparison **125** can utilize the importance (i.e. as indicated by importance index [later described], etc.) of Object Representations **625** or portions thereof for determining substantial similarity of Collections of Object Representations **525**. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to more important Object Representations **625** or portions thereof such as Object Representations **625** representing near Objects **615**, Object Representations **625** representing large Objects **615**, etc., thereby tolerating mismatches in less important Object Representations **625** or portions thereof such as Object Representations **625** representing distant Objects **615**, Object Representations **625** representing small Objects **615**, etc. In general, any Object Representation **625** or portion thereof can be assigned higher or lower importance. In further aspects, Similarity Comparison **125** can omit some of the Object Representations **625** or portions thereof from the comparison in determining substantial similarity of Collections of Object Representations **525**. In one example, Object Representations **625** representing distant Objects **615** can be omitted from comparison. In another example, Object Representations **625** representing small Objects **615** can be omitted from comparison. In general, any Object Representation **625** or portion thereof can be omitted from comparison depending on implementation.

Similarity Comparison **125** can automatically adjust (i.e. increase or decrease) the strictness of the rules for determining substantial similarity of Collections of Object Representations **525**. In some aspects, such adjustment in strictness can be done by Similarity Comparison **125** in response to determining that total equivalence of compared Collections of Object Representations **525** had not been found. Similarity Comparison **125** can keep adjusting the strictness rules until a substantial similarity is found. All the rules or settings of substantial similarity can be set, reset, or adjusted by Similarity Comparison **125** in response to another strictness level determination. For example, Similarity Comparison **125** may attempt to find a match or substantial match in a certain percentage (i.e. 81%, etc.) of Object Representations **625** or portions thereof from the compared Collections of Object Representations **525**. If the comparison does not determine substantial similarity of compared Collections of Object Representations **525**, Similarity Comparison **125** may decide to decrease the strictness of the rules. In response, Similarity Comparison **125** may attempt to find fewer matching or substantially matching Object Representations **625** or portions thereof than in the previous attempt using stricter rules. If the comparison still does not determine substantial similarity of compared Collections of Object Representations **525**, Similarity Comparison **125** may further decrease the strictness (i.e. down to a certain minimum strictness or threshold, etc.) by requiring fewer Object Representations **625** or portions thereof to match or substantially match, thereby further increasing a chance of finding substantial similarity in compared Collections of Object Representations **525**.

Where a reference to Object Representation **625** is used herein it should be understood that a portion of Object Representation **625** (i.e. Object Property **630**, etc.) or a plurality of Object Representations **625** can be used instead

of or in addition to the Object Representation **625**. In one example, instead of or in addition to Object Representation **625**, Object Properties **630** and/or other portions that constitute an Object Representation **625** can be compared. In another example, instead of or in addition to Object Representation **625**, plurality of Object Representations **625** can be compared. As such, any operations, rules, logic, and/or functions operating on Object Representation **625** may similarly apply to any portion of Object Representation **625** and/or a plurality of Object Representations **625** as applicable. In general, whole Object Representations **625**, portions of Object Representations **625**, and/or pluralities of Object Representations **625**, including any operations thereon, can be combined to arrive at desired results. Some or all of the above-described rules, logic, and/or techniques can be utilized alone or in combination with each other or with other rules, logic, and/or techniques. One of ordinary skill in art will recognize that other techniques known in art for determining similarity of Object Representations **625** and/or other data that would be too voluminous to describe are within the scope of this disclosure.

In some embodiments, in determining substantial similarity of Object Representations **625** (i.e. Object Representations **625** from the compared Collections of Object Representations **525**, etc.), Similarity Comparison **125** can compare Object Properties **630** or portions (i.e. characters, words, numbers, etc.) thereof from one Object Representation **625** with Object Properties **630** or portions thereof from another Object Representation **625**. In some aspects, total equivalence is found when all Object Properties **630** or portions thereof of one Object Representation **625** match all Object Properties **630** or portions thereof of another Object Representation **625**. In other aspects, if total equivalence is not found, Similarity Comparison **125** may attempt to determine substantial similarity of compared Object Representations **625**. In one example, substantial similarity can be achieved when most of the Object Properties **630** or portions thereof from the compared Object Representations **625** match or substantially match. In another example, substantial similarity can be achieved when at least a threshold number (i.e. 1, 2, 3, 6, 11, etc.) or percentage (i.e. 55%, 61%, 78%, 82%, 99%, etc.) of Object Properties **630** or portions thereof from the compared Object Representations **625** match or substantially match. Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching Object Properties **630** or portions thereof from the compared Object Representations **625** exceeds a threshold number (i.e. 1, 2, 3, 6, 11, etc.) or a threshold percentage (i.e. 55%, 61%, 78%, 82%, 99%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or percentage of Object Properties **630** or portions thereof from the compared Object Representations **625** match or substantially match. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In further aspects, Similarity Comparison **125** can utilize Categories **635** associated with Object Properties **630** for determining substantial similarity of Object Representations **625**. In one example, Object Properties **630** or portions thereof from the compared Object Representations **625** in a same Category **635** may be compared. This way, Object Properties **630** or portions thereof can be compared with their own peers. In one instance, Object Properties **630** or portions thereof from the compared Object Representations **625** in Category **635** "Type" may be compared. Any text comparison technique can be utilized in such comparing. In another instance, Object Properties **630** or portions thereof from the compared Object Representations **625** in Category **635** "Distance" or "Bearing" may be compared. Any number comparison technique can be utilized in such comparing. In a further instance, Object Properties **630** or portions thereof from the compared Object Representations **625** in Category **635** "Shape" may be compared. Any model, point cloud, or other computer construct comparison technique can be utilized in such comparing. In further aspects, Similarity Comparison **125** can utilize the importance (i.e. as indicated by importance index [later described], etc.) of Object Properties **630** or portions thereof for determining substantial similarity of Object Representations **625**. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to more important Object Properties **630** or portions thereof such as Object Properties **630** or portions thereof in Categories **635** "Type", "Distance", "Bearing", etc., thereby tolerating mismatches in less important Object Properties **630** or portions thereof such as Object Properties **630** or portions thereof in Categories **635** "Identity", "Shape", etc. In general, any Object Property **630** or portion thereof can be assigned higher or lower importance. In further aspects, Similarity Comparison **125** can omit some of the Object Properties **630** or portions thereof from the comparison in determining substantial similarity of Object Representations **625**. In one example, Object Properties **630** or portions thereof in Category **635** "Identity" can be omitted from comparison. In another example, Object Properties **630** or portions thereof in Category **635** "Shape" can be omitted from comparison. In general, any Object Property **630** or portion thereof can be omitted from comparison depending on implementation.

Similarity Comparison **125** can automatically adjust (i.e. increase or decrease) the strictness of the rules for determining substantial similarity of Object Representations **625**. In some aspects, such adjustment in strictness can be done by Similarity Comparison **125** in response to determining that total equivalence of compared Object Representations **625** had not been found. Similarity Comparison **125** can keep adjusting the strictness rules until a substantial similarity is found. All the rules or settings of substantial similarity can be set, reset, or adjusted by Similarity Comparison **125** in response to another strictness level determination. For example, Similarity Comparison **125** may attempt to find a match or substantial match in a certain percentage (i.e. 87%, etc.) of Object Properties **630** or portions thereof from the compared Object Representations **625**. If the comparison does not determine substantial similarity of compared Object Representations **625**, Similarity Comparison **125** may decide to decrease the strictness of the rules. In response, Similarity Comparison **125** may attempt to find fewer matching or substantially matching Object Properties **630** or portions thereof than in the previous attempt using stricter rules. If the comparison still does not determine substantial similarity of compared Object Representations **625**, Similarity Comparison **125** may further decrease the strictness (i.e. down to a certain minimum strictness or threshold, etc.) by requiring fewer Object Properties **630** or portions thereof to match or substantially match, thereby further increasing a chance of finding substantial similarity in compared Object Representations **625**. In further aspects, an adjustment in strictness can be done by Similarity Comparison **125** in response to determining that multiple substantially similar Object Representations **625** had been found. Similarity Comparison **125** can keep adjusting the strictness of the rules until a best of the substantially

similar Object Representations **625** is found. For example, Similarity Comparison **125** may attempt to find a match or substantial match in a certain percentage (i.e. 65%, etc.) of Object Properties **630** or portions thereof from the compared Object Representations **625**. If the comparison determines a number of substantially similar Object Representations **625**, Similarity Comparison **125** may decide to increase the strictness of the rules to decrease the number of substantially similar Object Representations **625**. In response, Similarity Comparison **125** may attempt to find more matching or substantially matching Object Properties **630** or portions thereof in addition to the earlier found Object Properties **630** or portions thereof to limit the number of substantially similar Object Representations **625**. If the comparison still provides more than one substantially similar Object Representation **625**, Similarity Comparison **125** may further increase the strictness by requiring additional Object Properties **630** or portions thereof to match or substantially match, thereby further narrowing the number of substantially similar Object Representations **625** until a best substantially similar Object Representation **625** is found.

Where a reference to Object Property **630** is used herein it should be understood that a portion of Object Property **630** or a plurality of Object Properties **630** can be used instead of or in addition to the Object Property **630**. In one example, instead of or in addition to Object Property **630**, characters, words, numbers, and/or other portions that constitute an Object Property **630** can be compared. In another example, instead of or in addition to Object Property **630**, a plurality of Object Properties **630** can be compared. As such, any operations, rules, logic, and/or functions operating on Object Property **630** may similarly apply to any portion of Object Property **630** and/or a plurality of Object Properties **630** as applicable. In general, whole Object Properties **630**, portions of Object Properties **630**, and/or pluralities of Object Properties **630**, including any operations thereon, can be combined to arrive at desired results. Some or all of the above-described rules, logic, and/or techniques can be utilized alone or in combination with each other or with other rules, logic, and/or techniques. One of ordinary skill in art will recognize that other techniques known in art for determining similarity of Object Properties **630** and/or other data that would be too voluminous to describe are within the scope of this disclosure.

In some embodiments where compared Knowledge Cells **800** include a stream of Collections of Object Representations **525**, in determining similarity of Knowledge Cells **800**, Similarity Comparison **125** can perform collective comparison of Collections of Object Representations **525** or portions (i.e. Object Representations **625**, Object Properties **630**, etc.) thereof such as comparison of a stream of Collections of Object Representations **525** or portions thereof from one Knowledge Cell **800** with a stream of Collections of Object Representations **525** or portions thereof from another Knowledge Cell **800**. Similarity Comparison **125** of collectively compared Collections of Object Representations **525** or portions thereof may include any features, functionalities, and embodiments of the previously described Similarity Comparison **125** of individually compared Collections of Object Representations **525** or portions thereof. In some aspects, total equivalence is found when all Collections of Object Representations **525** or portions thereof from one Knowledge Cell **800** match all Collections of Object Representations **525** or portions thereof from another Knowledge Cell **800**. If total equivalence is not found, Similarity Comparison **125** may attempt to determine substantial or other similarity of compared Knowledge Cells **800**. In one example, substantial similarity can be achieved when most of the Collections of Object Representations **525** or portions thereof from the compared Knowledge Cells **800** match or substantially match. In another example, substantial similarity can be achieved when at least a threshold number (i.e. 1, 2, 4, 9, 33, 138, etc.) or percentage (i.e. 39%, 58%, 77%, 88%, 94%, etc.) of Collections of Object Representations **525** or portions thereof from the compared Knowledge Cells **800** match or substantially match. Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching Collections of Object Representations **525** or portions thereof from the compared Knowledge Cells **800** exceeds a threshold number (i.e. 1, 2, 4, 9, 33, 138, etc.) or a threshold percentage (i.e. 39%, 58%, 77%, 88%, 94%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or percentage of Collections of Object Representations **525** or portions thereof from the compared Knowledge Cells **800** match or substantially match. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In some aspects, Similarity Comparison **125** can utilize the importance (i.e. as indicated by importance index [later described], etc.) of Collections of Object Representations **525** or portions thereof for determining substantial similarity of Knowledge Cells **800**. In one example, substantial similarity can be achieved when matches or substantial matches are found with respect to more important Collections of Object Representations **525** or portions thereof such as more substantive or larger Collections of Object Representations **525** (i.e. Collections of Object Representations **525** comprising a higher number of Object Representations **625**, etc.) or portions thereof, etc., thereby tolerating mismatches in less important Collections of Object Representations **525** or portions thereof such as less substantive or smaller Collections of Object Representations **525** (i.e. Collections of Object Representations **525** comprising a lower number of Object Representations **625**, etc.) or portions thereof, etc. In general, any Collection of Object Representations **525** or portion thereof can be assigned higher or lower importance. In other aspects, Similarity Comparison **125** can utilize the order of Collections of Object Representations **525** or portions thereof for determining substantial similarity of Knowledge Cells **800**. In one example, substantial similarity can be achieved when matches or substantial matches are found in earlier Collections of Object Representations **525** or portions thereof from the compared Knowledge Cells **800**, thereby tolerating mismatches in later Collections of Object Representations **525** or portions thereof. In another example, substantial similarity can be achieved when matches or substantial matches are found in corresponding (i.e. similarly ordered, temporally related, etc.) Collections of Object Representations **525** or portions thereof from the compared Knowledge Cells **800**. In one instance, a 94th Collection of Object Representations **525** or portions thereof from one Knowledge Cell **800** can be compared with a 94th Collection of Object Representations **525** or portions thereof from another Knowledge Cell **800**. In another instance, a 94th Collection of Object Representations **525** or portions thereof from one Knowledge Cell **800** can be compared with a number of Collections of Object Representations **525** or portions thereof around (i.e. preceding and/or following) a 94th Collection of Object Representations **525** from another Knowledge Cell **800**. This way, flexibility can be implemented in finding a substantially similar Collection of

Object Representations **525** or portions thereof if the Collections of Object Representations **525** or portions thereof in the compared Knowledge Cells **800** are not perfectly aligned. In a further instance, Similarity Comparison **125** can utilize Dynamic Time Warping (DTW) and/or other techniques known in art for comparing and/or aligning temporal sequences (i.e. streams of Collections of Object Representations **525** or portions thereof, etc.) that may vary in time or speed. In further aspects, Similarity Comparison **125** can omit some of the Collections of Object Representations **525** or portions thereof from the comparison in determining substantial similarity of Knowledge Cells **800**. In one example, less substantive or smaller Collections of Object Representations **525** or portions thereof can be omitted from comparison. In another example, some or all Collections of Object Representations **525** or portions thereof related to a specific time period can be omitted from comparison. In general, any Collection of Object Representations **525** or portion thereof can be omitted from comparison depending on implementation.

Similarity Comparison **125** can automatically adjust (i.e. increase or decrease) the strictness of the rules for determining substantial similarity of Knowledge Cells **800**. In some aspects, such adjustment in strictness can be done by Similarity Comparison **125** in response to determining that total equivalence of compared Knowledge Cells **800** had not been found. Similarity Comparison **125** can keep adjusting the strictness of the rules until substantial similarity is found. All the rules or settings of substantial similarity can be set, reset, or adjusted by Similarity Comparison **125** in response to another strictness level determination. For example, Similarity Comparison **125** may attempt to find a match or substantial match in a certain percentage (i.e. 92%, etc.) of Collections of Object Representations **525** or portions thereof from the compared Knowledge Cells **800**. If the comparison does not determine substantial similarity of compared Knowledge Cells **800**, Similarity Comparison **125** may decide to decrease the strictness of the rules. In response, Similarity Comparison **125** may attempt to find fewer matching or substantially matching Collections of Object Representations **525** or portions thereof than in the previous attempt using stricter rules. If the comparison still does not determine substantial similarity of compared Knowledge Cells **800**, Similarity Comparison **125** may further decrease (i.e. down to a certain minimum strictness or threshold, etc.)

the strictness by requiring fewer Collections of Object Representations **525** or portions thereof to match or substantially match, thereby further increasing a chance of finding substantial similarity in compared Knowledge Cells **800**. In further aspects, an adjustment in strictness can be done by Similarity Comparison **125** in response to determining that multiple substantially similar Knowledge Cells **800** had been found. Similarity Comparison **125** can keep adjusting the strictness of the rules until a best of the substantially similar Knowledge Cells **800** is found. For example, Similarity Comparison **125** may attempt to find a match or substantial match in a certain percentage (i.e. 71%, etc.) of Collections of Object Representations **525** or portions thereof from the compared Knowledge Cells **800**. If the comparison determines a number of substantially similar Knowledge Cells **800**, Similarity Comparison **125** may decide to increase the strictness of the rules to decrease the number of substantially similar Knowledge Cells **800**. In response, Similarity Comparison **125** may attempt to find more matching or substantially matching Collections of Object Representations **525** or portions thereof in addition to the earlier found Collections of Object Representations **525** or portions thereof to limit the number of substantially similar Knowledge Cells **800**. If the comparison still provides more than one substantially similar Knowledge Cell **800**, Similarity Comparison **125** may further increase the strictness by requiring additional Collections of Object Representations **525** or portions thereof to match or substantially match, thereby further narrowing the number of substantially similar Knowledge Cells **800** until a best substantially similar Knowledge Cell **800** is found.

Some or all of the aforementioned rules, logic, and/or techniques for determining substantial similarity of Knowledge Cells **800** can be utilized alone or in combination with each other or with other rules, logic, and/or techniques. One of ordinary skill in art will recognize that other techniques known in art for determining similarity of Knowledge Cells **800** and/or other data that would be too voluminous to describe are within the scope of this disclosure.

In any of the comparisons involving numbers such as, for example, Object Properties **630** including numbers (i.e. distances, bearings/angles, etc.), Similarity Comparison **125** can compare a number from one Object Property **630** with a number from another Object Property **630**. In some aspects, total equivalence is found when the number from one Object Property **630** equals the number from another Object Property **630**. In other aspects, if total equality is not found, Similarity Comparison **125** may attempt to determine substantial similarity of the compared numbers using a tolerance or threshold for determining a match. In some aspects, Similarity Comparison **125** can utilize a threshold for acceptable number difference in determining a match of compared numbers. For example, a threshold for acceptable number difference (i.e. absolute difference, etc.) can be set at 10. Therefore, **130** matches or is sufficiently similar to **135** because the number difference (i.e. 5 in this example) is lower than the threshold for acceptable number difference (i.e. 10 in this example, etc.). Furthermore, **130** does not match or is not sufficiently similar to **143** because the number difference (i.e. 13 in this example) is greater than the threshold for acceptable number difference. Any other threshold for acceptable number difference can be used such as 0.024, 1, 8, 15, 77, 197, 2438, 728322, and/or others. In other aspects, Similarity Comparison **125** can utilize a threshold for acceptable percentage difference in determining a match of compared numbers. For example, a threshold for acceptable percentage difference can be set at 10%. Therefore, 100 matches or is sufficiently similar to 106 because the percentage difference (i.e. 6% in this example) is lower than the threshold for acceptable percentage difference (i.e. 10% in this example). Furthermore, 100 does not match or is not sufficiently similar to 84 because the percentage difference (i.e. 16% in this example) is higher than the threshold for acceptable percentage difference. Any other threshold for acceptable percentage difference can be used such as 0.68%, 1%, 3%, 11%, 33%, 69%, 87%, and/or others. The aforementioned thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. Other techniques known in art for comparing numbers can be utilized herein.

In any of the comparisons involving text such as, for example, Object Properties **630** including text (i.e. types, identities, etc.), Similarity Comparison **125** can compare words, characters, and/or other text from one Object Property **630** with words, characters, and/or other text from another Object Property **630**. In some aspects, total equiva-

lence is found when all words, characters, and/or other text from one Object Property **630** match all words, characters, and/or other text from another Object Property **630**. In other aspects, if total equivalence is not found, Similarity Comparison **125** may attempt to determine substantial similarity of compared Object Properties **630**. In one example, substantial similarity can be achieved when most of the words, characters, and/or other text from the compared Object Properties **630** match or substantially match. In another example, substantial similarity can be achieved when at least a threshold number (i.e. 1, 2, 3, 4, 7, 11, etc.) or percentage (i.e. 38%, 63%, 77%, 84%, 98%, etc.) of words, characters, and/or other text from the compared Object Properties **630** match or substantially match.

Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching words, characters, and/or other text from the compared Object Properties **630** exceeds a threshold number (i.e. 1, 2, 3, 4, 7, 11, etc.) or a threshold percentage (i.e. 48%, 63%, 77%, 84%, 98%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or percentage of words, characters, and/or other text from the compared Object Properties **630** match or substantially match. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In further aspects, Similarity Comparison **125** can utilize the importance (i.e. as indicated by importance index [later described], etc.) of words, characters, and/or other text for determining substantial similarity of Object Properties **630**. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to more important words, characters, and/or other text such as longer words and/or other text, thereby tolerating mismatches in less important words, characters, and/or other text such as shorter words and/or other text. In general, any word, character, and/or other text can be assigned higher or lower importance. In further aspects, Similarity Comparison **125** can utilize the order of words, characters, and/or other text for determining substantial similarity of Object Properties **630**. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to front-most words, characters, and/or other text, thereby tolerating mismatches in later words, characters, and/or other text. In further aspects, Similarity Comparison **125** can utilize semantic conversion to account for variations of words and/or other text. In one example, Object Property **630** may include a word "house". In addition to searching for the exact word in a compared Object Property **630**, Similarity Comparison **125** can employ semantic conversion and attempt to match "home", "residence", "dwelling", "place", or other semantically similar variations of the word with a meaning "house". In another example, Object Property **630** may include a word "buy". In addition to searching for the exact word in a compared Object Property **630**, Similarity Comparison **125** can employ semantic conversion and attempt to match "buying", "bought", or other semantically similar variations of the word with a meaning "buy" in different tenses. Any other grammatical analysis or transformation can be utilized to cover the full scope of word and/or other text variations. In some designs, semantic conversion can be implemented using a thesaurus or dictionary. In another example, semantic conversion can be implemented using a table where each row comprises semantically similar variations of a word and/or other text. In further aspects, Similarity Comparison **125** can utilize a language model for understanding or interpreting the concepts contained in the words and/or other text and compare the concepts instead of or in addition to the words and/or other text. Examples of language models include unigram model, n-gram model, neural network language model, bag of words model, and/or others. Any of the techniques for matching of words can similarly be used for matching of concepts. In further aspects, Similarity Comparison **125** can omit some of the words, characters, and/or other text from the comparison in determining substantial similarity of Object Properties **630**. In one example, rear-most words, characters, and/or other text can be omitted from comparison. In another example, shorter words and/or other text can be omitted from comparison. In general, any word, character, and/or other text can be omitted from comparison depending on implementation. Other techniques known in art for comparing words, characters, and/or other text can be utilized herein.

In some embodiments, Similarity Comparison **125** can compare one or more Extra Info **527** (i.e. time information, location information, computed information, contextual information, and/or other information, etc.) in addition to or instead of comparing Collections of Object Representations **525** or portions thereof in determining substantial similarity of Knowledge Cells **800**. Extra Info **527** can be set to be less, equally, or more important (i.e. as indicated by importance index [later described], etc.) than Collections of Object Representations **525**, Object Representations **625**, Object Properties **630**, and/or other elements in the comparison. Since Extra Info **527** may include any contextual or other information that can be useful in determining similarity of any compared elements, Extra Info **527** can be used to enhance any of the aforementioned similarity determinations as applicable.

In some embodiments, Similarity Comparison **125** can also compare one or more Instruction Sets **526** in addition to or instead of comparing Collections of Object Representations **525** or portions thereof in determining substantial similarity of Knowledge Cells **800**. In some aspects, Similarity Comparison **125** can compare portions of Instruction Sets **526** to determine substantial or other similarity of Instruction Sets **526**. Similar to the above-described thresholds, thresholds for the number or percentage of matching portions of the compared Instruction Sets **526** can be utilized in determining substantial or other similarity of the compared Instruction Sets **526**. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In other aspects, Similarity Comparison **125** can compare text (i.e. characters, words, phrases, etc.), numbers, or other data (i.e. bits, etc.) to determine substantial or other similarity of Instruction Sets **526**. Any other comparison technique can be utilized in comparing Instruction Sets **526** in alternate implementations. Instruction Sets **526** can be set to be less, equally, or more important (i.e. as indicated by importance index [later described], etc.) than Collections of Object Representations **525**, Object Representations **625**, Object Properties **630**, Extra Info **527**, and/or other elements in the comparison.

In some embodiments, an importance index (not shown) or other importance ranking technique can be used in any of the previously described comparisons or other processing involving elements of different importance. Importance index indicates importance of the element to or with which the index is assigned or associated. For example, importance index may indicate importance of a Knowledge Cell **800**,

Collection of Object Representations **525**, Object Representation **625**, Object Property **630**, Instruction Set **526**, Extra Info **527**, and/or other element to or with which the index is assigned or associated. In some aspects, importance index on a scale from 0 to 1 can be utilized, although, any other range can also be utilized. Importance index can be stored in or associated with the element to which the index pertains. Importance indexes of various elements can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. In one example, a higher Importance index can be assigned to more substantive or larger Collections of Object Representations **525** (i.e. Collections of Object Representations **525** comprising a higher number of Object Representations **625**, etc.). In another example, a higher importance index can be assigned to Object Representations **625** representing closer, larger, and/or other Objects **615**. Any importance index can be assigned to or associated with any element described herein depending on implementation. Any importance ranking technique can be utilized as or instead of importance index in alternate embodiments.

In some embodiments, Similarity Comparison **125** may generate a similarity index (not shown) for any of the compared elements. Similarity index indicates how well an element is matched with another element. For example, similarity index indicates how well a Knowledge Cell **800**, Collection of Object Representations **525**, Object Representation **625**, Object Property **630**, Instruction Set **526**, Extra Info **527**, and/or other element is matched with a compared element. In some aspects, similarity index on a scale from 0 to 1 can be utilized, although, any other range can also be utilized. Similarity index can be generated by Similarity Comparison **125** whether substantial or other similarity between the compared elements is achieved or not. In one example, similarity index can be determined for a Knowledge Cell **800** based on a ratio/percentage of matched or substantially matched Collections of Object Representations **525** relative to the number of Collections of Object Representations **525** in the compared Knowledge Cell **800**. Specifically, similarity index of 0.91 is determined if 91% of Collections of Object Representations **525** of one Knowledge Cell **800** match or substantially match Collections of Object Representations **525** of another Knowledge Cell **800**. In some designs, importance (i.e. as indicated by importance index, etc.) of one or more Collections of Object Representations **525** can be included in the calculation of a weighted similarity index. Similar determination of similarity index can be implemented with Collections of Object Representations **525**, Object Representations **625**, Object Properties **630**, Instruction Sets **526**, Extra Info **527**, and/or other elements or portions thereof. Any combination of the aforementioned similarity index determinations or calculations can be utilized in alternate embodiments. Any similarity ranking technique can be utilized to determine or calculate similarity index in alternate embodiments.

Referring to FIG. **21**, an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Neural Network **530*a*** comprising shortcut Connections **853** is illustrated. In some designs, Knowledge Cells **800** in one Layer **854** of Neural Network **530*a*** can be connected with Knowledge Cells **800** in any Layer **854**, not only in a successive Layer **854**, thereby creating shortcuts (i.e. shortcut Connections **853**, etc.) through Neural Network **530*a***. In some aspects, creating a shortcut Connection **853** can be implemented by performing Similarity Comparisons **125** of a Knowledge Cell **800** from Knowledge Structuring Unit **520** with Knowledge Cells **800** in any Layer **854** when applying (i.e. storing, copying, etc.) the Knowledge Cell **800** from Knowledge Structuring Unit **520** onto Neural Network **530*a***. Once created, shortcut Connections **853** enable a wider variety of Knowledge Cells **800** to be considered when selecting a path through Neural Network **530*a***. In some embodiments, Knowledge Structuring Unit **520** structures or generates Knowledge Cells **800** and the system applies them onto Neural Network **530*a***, thereby implementing learning Device's **98** operation in circumstances including objects with various properties. The system can perform Similarity Comparisons **125** of a Knowledge Cell **800** from Knowledge Structuring Unit **520** with Knowledge Cells **800** in one or more Layers **854** of Neural Network **530*a***. If a substantially similar Knowledge Cell **800** is not found in the one or more Layers **854** of Neural Network **530*a***, the system may insert (i.e. copy, store, etc.) the Knowledge Cell **800** from Knowledge Structuring Unit **520** into a Layer **854** of Neural Network **530*a***, and create a Connection **853** to the inserted Knowledge Cell **800** from a prior Knowledge Cell **800** including assigning an occurrence count to the new Connection **853**, calculating a weight of the new Connection **853**, and updating any other Connections **853** originating from the prior Knowledge Cell **800**. On the other hand, if a substantially similar Knowledge Cell **800** is found in the one or more Layers **854** of Neural Network **530*a***, the system may update occurrence count and weight of Connection **853** to that Knowledge Cell **800** from a prior Knowledge Cell **800**, and update any other Connections **853** originating from the prior Knowledge Cell **800**. Any of the previously described and/or other techniques for comparing, inserting, updating, and/or other operations on Knowledge Cells **800**, Connections **853**, Layers **854**, and/or other elements can similarly be utilized in Neural Network **530*a*** that comprises shortcut Connections **853**.

Referring to FIG. **22**, an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Graph **530*b*** is illustrated. In some aspects, any Knowledge Cell **800** can be connected with any other Knowledge Cell **800** in Graph **530*b***. In other aspects, any Knowledge Cell **800** can be connected with itself and/or any other Knowledge Cell **800** in Graph **530*b***. In some embodiments, Knowledge Structuring Unit **520** structures or generates Knowledge Cells **800** and the system applies (i.e. store, copy, etc.) them onto Graph **530*b***, thereby implementing learning Device's **98** operation in circumstances including objects with various properties. The system can perform Similarity Comparisons **125** of a Knowledge Cell **800** from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Graph **530*b***. If a substantially similar Knowledge Cell **800** is not found in Graph **530*b***, the system may insert (i.e. copy, store, etc.) the Knowledge Cell **800** from Knowledge Structuring Unit **520** into Graph **530*b***, and create a Connection **853** to the inserted Knowledge Cell **800** from a prior Knowledge Cell **800** including assigning an occurrence count to the new Connection **853**, calculating a weight of the new Connection **853**, and updating any other Connections **853** originating from the prior Knowledge Cell **800**. On the other hand, if a substantially similar Knowledge Cell **800** is found in Graph **530*b***, the system may update occurrence count and weight of Connection **853** to that Knowledge Cell **800** from a prior Knowledge Cell **800**, and update any other Connections **853** originating from the prior Knowledge Cell **800**. Any of the

previously described and/or other techniques for comparing, inserting, updating, and/or other operations on Knowledge Cells **800**, Connections **853**, and/or other elements can similarly be utilized in Graph **530***b*.

For example, the system can perform Similarity Comparisons **125** of Knowledge Cell **800***ba* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Graph **530***b*. In the case that a substantially similar match is not found, the system may insert Knowledge Cell **800***ha* into Graph **530***b* and copy Knowledge Cell **800***ba* into the inserted Knowledge Cell **800***ha*. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bb* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Graph **530***b*. In the case that a substantially similar match is found between Knowledge Cell **800***bb* and Knowledge Cell **800***hb*, the system may create Connection **853***h***1** between Knowledge Cell **800***ha* and Knowledge Cell **800***hb* with occurrence count of 1 and weight of 1. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bc* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Graph **530***b*. In the case that a substantially similar match is found between Knowledge Cell **800***bc* and Knowledge Cell **800***hc*, the system may update occurrence count and weight of Connection **853***h***2** between Knowledge Cell **800***hb* and Knowledge Cell **800***hc*, and update weights of other outgoing Connections **853** (one in this example) originating from Knowledge Cell **800***hb* as previously described. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bd* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Graph **530***b*. In the case that a substantially similar match is not found, the system may insert Knowledge Cell **800***hd* into Graph **530***b* and copy Knowledge Cell **800***bd* into the inserted Knowledge Cell **800***hd*. The system may also create Connection **853***h***3** between Knowledge Cell **800***hc* and Knowledge Cell **800***hd* with occurrence count of 1 and weight calculated based on the occurrence count as previously described. The system may also update weights of other outgoing Connections **853** (one in this example) originating from Knowledge Cell **800***hc* as previously described. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***be* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Graph **530***b*. In the case that a substantially similar match is not found, the system may insert Knowledge Cell **800***he* into Graph **530***b* and copy Knowledge Cell **800***be* into the inserted Knowledge Cell **800***he*. The system may also create Connection **853***h***4** between Knowledge Cell **800***hd* and Knowledge Cell **800***he* with occurrence count of 1 and weight of 1. Applying any additional Knowledge Cells **800** from Knowledge Structuring Unit **520** onto Graph **530***b* follows similar logic or process as the above-described.

Referring to FIG. **23**, an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Collection of Sequences **530***c* is illustrated. Collection of Sequences **530***c* comprises the functionality for storing one or more Sequences **533**. Sequence **533** comprises the functionality for storing any number of Knowledge Cells **800**. For example, Knowledge Structuring Unit **520** structures or generates Knowledge Cells **800** and the system applies them onto Collection of Sequences **530***c*, thereby implementing learning Device's **98** operation in circumstances including objects with various properties. The system can perform collective Similarity Comparisons **125** of Knowledge Cells **800** from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Sequences **533** of Collection of Sequences **530***c* to find a Sequence **533** comprising Knowledge Cells **800** that are collectively substantially similar to the Knowledge Cells **800** from Knowledge Structuring Unit **520**. If Sequence **533** comprising such collectively substantially similar Knowledge Cells **800** is not found in Collection of Sequences **530***c*, the system may create a new Sequence **533** comprising the Knowledge Cells **800** from Knowledge Structuring Unit **520** and insert (i.e. copy, store, etc.) the new Sequence **533** into Collection of Sequences **530***c*. On the other hand, if Sequence **533** comprising collectively substantially similar Knowledge Cells **800** is found in Collection of Sequences **530***c*, the system may optionally omit inserting the Knowledge Cells **800** from Knowledge Structuring Unit **520** into Collection of Sequences **530***c* as inserting a similar Sequence **533** may not add much or any additional knowledge. This approach can save storage resources and limit the number of Knowledge Cells **800** that may later need to be processed or compared. In some aspects, a Sequence **533** may include Knowledge Cells **800** relating to a single operation of Device **98**. In other aspects, a Sequence **533** may include Knowledge Cells **800** relating to a part of an operation of Device **98**. In further aspects, one or more long Sequences **533** each including Knowledge Cells **800** of multiple operations of Device **98** can be utilized. In one example, Knowledge Cells **800** of all operations can be stored in a single long Sequence **533** in which case Collection of Sequences **530***c* as a separate element can be omitted. In another example, Knowledge Cells **800** of multiple operations can be included in a plurality of long Sequences **533** such as hourly, daily, weekly, monthly, yearly, or other periodic or other Sequences **533**. Similarity Comparisons **125** can be performed by traversing the one or more long Sequences **533** to find a match or substantially similar match. For instance, the system can perform collective Similarity Comparisons **125** of Knowledge Cells **800** from Knowledge Structuring Unit **520** with Knowledge Cells **800** in subsequences of a long Sequence **533** in incremental or other traversing pattern to find a subsequence comprising Knowledge Cells **800** that are collectively substantially similar to the Knowledge Cells **800** from Knowledge Structuring Unit **520**. The incremental traversing pattern may start from one end of a long Sequence **533** and move the comparison subsequence up or down one or any number of incremental Knowledge Cells **800** at a time. Other traversing patterns or methods can be employed such as starting from the middle of the Sequence **533** and subdividing the resulting sub-sequences in a recursive pattern, or any other traversing pattern or method. If a subsequence comprising collectively substantially similar Knowledge Cells **800** is not found in the long Sequence **533**, the system may concatenate or append the Knowledge Cells **800** from Knowledge Structuring Unit **520** to the long Sequence **533**. In further aspects, Connections **853** can optionally be used in Sequence **533** to connect Knowledge Cells **800**. For example, a Knowledge Cell **800** can be connected not only with a next Knowledge Cell **800** in the Sequence **533**, but also with any other Knowledge Cell **800** in the Sequence **533**, thereby creating alternate routes or shortcuts through the Sequence **533**. Any number of Connections **853** connecting any Knowledge Cells **800** can be utilized. Any of the previously described and/or other techniques for comparing, inserting, updating, and/or other operations on Knowledge Cells **800**, Connections **853**, and/or other elements can similarly be utilized in Sequences **533** and/or Collection of Sequences **530***c*.

In some embodiments, various elements and/or techniques can be utilized in the aforementioned substantial similarity determinations with respect to collectively compared Knowledge Cells **800** and/or other elements. In some aspects, substantial similarity of collectively compared Knowledge Cells **800** can be determined based on similarities or similarity indexes of the individually compared Knowledge Cells **800**. In one example, an average of similarities or similarity indexes of individually compared Knowledge Cells **800** can be used to determine similarity of collectively compared Knowledge Cells **800**. In another example, a weighted average of similarities or similarity indexes of individually compared Knowledge Cells **800** can be used to determine similarity of collectively compared Knowledge Cells **800**. For instance, to affect the weighting of collective similarity, a higher weight or importance (i.e. importance index, etc.) can be assigned to the similarities or similarity indexes of some Knowledge Cells **800** and lower for other Knowledge Cells **800**. Any higher or lower weight or importance assignment can be implemented. In other aspects, any of the previously described or other thresholds for substantial similarity of individually compared elements can similarly be utilized for collectively compared elements. In one example, substantial similarity of collectively compared Knowledge Cells **800** can be achieved when their collective similarity or similarity index exceeds a similarity threshold. In another example, substantial similarity of collectively compared Knowledge Cells **800** can be achieved when at least a threshold number or percentage of Knowledge Cells **800** from the collectively compared Knowledge Cells **800** match or substantially match. Similarly, substantial similarity of collectively compared Knowledge Cells **800** can be achieved when a number or percentage of matching or substantially matching Knowledge Cells **800** from the collectively compared Knowledge Cells **800** exceeds a threshold. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. Similar elements and/or techniques as the aforementioned can be used for similarity determinations of other collectively compared elements such as Collections of Object Representations **525**, Object Representations **625**, Object Properties **630**, Instruction Sets **526**, Extra Info **527**, and/or others. Similarity determinations of collectively compared elements may include any features, functionalities, and embodiments of Similarity Comparison **125**, and vice versa.

Any of the previously described data structures or arrangements of Knowledge Cells **800** such as Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, and/or others can be used alone, or in combination with each other or with other elements, in alternate embodiments. In one example, a path in Neural Network **530***a* or Graph **530***b* may include its own separate sequence of Knowledge Cells **800** that are not interconnected with Knowledge Cells **800** in other paths. In another example, a part of a path in Neural Network **530***a* or Graph **530***b* may include a sequence of Knowledge Cells **800** interconnected with Knowledge Cells **800** in other paths, whereas, another part of the path may include its own separate sequence of Knowledge Cells **800** that are not interconnected with Knowledge Cells **800** in other paths. Any other combinations or arrangements of Knowledge Cells **800** can be implemented.

Referring to FIG. **24**, an embodiment of determining anticipatory Instruction Sets **526** from a single Knowledge Cell **800** is illustrated. Knowledge Cell **800** may be part of a Knowledgebase **530** (i.e. Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.) such as Collection of Knowledge Cells **530***d*. Decision-making Unit **540** comprises the functionality for anticipating or determining a device's operation in circumstances including objects with various properties. Decision-making Unit **540** comprises the functionality for anticipating or determining Instruction Sets **526** to be used or executed in Device's **98** autonomous operation. In some aspects, Instruction Sets **526** anticipated or determined to be used or executed in Device's **98** autonomous operation may be referred to as anticipatory Instruction Sets **526**, alternate Instruction Sets **526**, and/or other suitable name or reference. Therefore, these terms can be used interchangeably herein depending on context. Decision-making Unit **540** also comprises other disclosed functionalities.

In some aspects, Decision-making Unit **540** may anticipate or determine Instruction Sets **526** (i.e. anticipatory Instruction Sets **526**, etc.) for autonomous Device **98** operation by performing Similarity Comparisons **125** of incoming Collections of Object Representations **525** or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Knowledgebase **530** (i.e. Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.). A Knowledge Cell **800** includes knowledge (i.e. one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**, etc.) of how Device **98** operated in a circumstance including objects with various properties as previously described. When one or more Collections of Object Representations **525** representing objects with similar properties are received in the future, Decision-making Unit **540** can anticipate the Instruction Sets **526** (i.e. anticipatory Instruction Sets **526**, etc.) previously learned in a similar circumstance, thereby enabling autonomous Device **98** operation. In some aspects, Decision-making Unit **540** can perform Similarity Comparisons **125** of incoming Collections of Object Representations **525** from Object Processing Unit **93** with Collections of Object Representations **525** from Knowledge Cells **800** in Knowledgebase **530** (i.e. Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.). If one or more substantially similar Collections of Object Representations **525** or portions thereof are found in a Knowledge Cell **800** from Knowledgebase **530**, Instruction Sets **526** (i.e. anticipatory Instruction Sets **526**, etc.) for autonomous Device **98** operation can be anticipated in Instruction Sets **526** correlated with the one or more Collections of Object Representations **525** from the Knowledge Cell **800**. In some designs, subsequent one or more Instruction Sets **526** for autonomous Device **98** operation can be anticipated in Instruction Sets **526** correlated with subsequent Collections of Object Representations **525** from the Knowledge Cell **800** or other Knowledge Cells **800**, thereby anticipating not only current, but also additional future Instruction Sets **526**. Although, Extra Info **527** is not shown in this and/or other figures for clarity of illustration, it should be noted that any Collection of Object Representations **525**, Instruction Set **526**, and/or other element may include or be associated with Extra Info **527** and that Decision-making Unit **540** can utilize Extra Info **527** for enhanced decision making.

For example, Decision-making Unit **540** can perform Similarity Comparison **125** of Collection of Object Representations **525/1** or portions thereof from Object Processing

Unit **93** with Collection of Object Representations **525*a*1** or portions thereof from Knowledge Cell **800*oa***. Collection of Object Representations **525*a*1** or portions thereof from Knowledge Cell **800*oa*** may be found substantially similar. Decision-making Unit **540** can anticipate Instruction Sets **526*a*1-526*a*3** correlated with Collection of Object Representations **525*a*1**, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparison **125** of Collection of Object Representations **525/2** or portions thereof from Object Processing Unit **93** with Collection of Object Representations **525*a*2** or portions thereof from Knowledge Cell **800*oa***. Collection of Object Representations **525*a*2** or portions thereof from Knowledge Cell **800*oa*** may be found substantially similar. Decision-making Unit **540** can anticipate Instruction Set **526*a*4** correlated with Collection of Object Representations **525*a*2**, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparison **125** of Collection of Object Representations **525/3** or portions thereof from Object Processing Unit **93** with Collection of Object Representations **525*a*3** or portions thereof from Knowledge Cell **800*oa***. Collection of Object Representations **525*a*3** or portions thereof from Knowledge Cell **800*oa*** may be found substantially similar. Decision-making Unit **540** may not anticipate any Instruction Sets **526** since none are correlated with Collection of Object Representations **525*a*3**. Decision-making Unit **540** can then perform Similarity Comparison **125** of Collection of Object Representations **525/4** or portions thereof from Object Processing Unit **93** with Collection of Object Representations **525*a*4** or portions thereof from Knowledge Cell **800*oa***. Collection of Object Representations **525*a*4** or portions thereof from Knowledge Cell **800*oa*** may not be found substantially similar. Decision-making Unit **540** can then perform Similarity Comparison **125** of Collection of Object Representations **525/5** or portions thereof from Object Processing Unit **93** with Collection of Object Representations **525*a*5** or portions thereof from Knowledge Cell **800*oa***. Collection of Object Representations **525*a*5** or portions thereof from Knowledge Cell **800*oa*** may not be found substantially similar. Decision-making Unit **540** can implement similar logic or process for any additional Collections of Object Representations **525** from Object Processing Unit **93**, and so on.

It should be understood that any of the described elements and/or techniques in the foregoing example can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells **800** or elements (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof would be affected accordingly. In one example, Extra Info **527** can be included in the Similarity Comparisons **125** as previously described. In another example, as history of incoming Collections of Object Representations **525** becomes available, Decision-making Unit **540** can perform collective Similarity Comparisons **125** of the history of Collections of Object Representations **525** or portions thereof from Object Processing Unit **93** with subsequences of Collections of Object Representations **525** or portions thereof from Knowledge Cell **800**. In a further example, the described comparisons in a single Knowledge Cell **800** may be performed on any number of Knowledge Cells **800** sequentially or in parallel. Parallel processors such as a plurality of Processors **11** or cores thereof can be utilized for such parallel processing. In a further example, various arrangements of Collections of Object Representations **525** and/or other elements in a Knowledge Cell **800** can be utilized as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to FIG. **25**, an embodiment of determining anticipatory Instruction Sets **526** by traversing a single Knowledge Cell **800** is illustrated. Knowledge Cell **800** may be part of a Knowledgebase **530** (i.e. Neural Network **530*a***, Graph **530*b***, Collection of Sequences **530*c***, Sequence **533**, Collection of Knowledge Cells **530*d***, etc.) such as Collection of Knowledge Cells **530*d***. For example, Decision-making Unit **540** can perform Similarity Comparison **125** of Collection of Object Representations **525/1** or portions thereof from Object Processing Unit **93** with Collection of Object Representations **525*a*1** or portions thereof from Knowledge Cell **800*oa***. Collection of Object Representations **525*a*1** or portions thereof from Knowledge Cell **800*oa*** may not be found substantially similar. Decision-making Unit **540** can then perform Similarity Comparison **125** of Collection of Object Representations **525/1** or portions thereof from Object Processing Unit **93** with Collection of Object Representations **525*a*2** or portions thereof from Knowledge Cell **800*oa***. Collection of Object Representations **525*a*2** or portions thereof from Knowledge Cell **800*oa*** may not be found substantially similar. Decision-making Unit **540** can then perform Similarity Comparison **125** of Collection of Object Representations **525/1** or portions thereof from Object Processing Unit **93** with Collection of Object Representations **525*a*3** or portions thereof from Knowledge Cell **800*oa***. Collection of Object Representations **525*a*3** or portions thereof from Knowledge Cell **800*oa*** may be found substantially similar. Decision-making Unit **540** may not anticipate any Instruction Sets **526** since none are correlated with Collection of Object Representations **525*a*3**. Decision-making Unit **540** can then perform Similarity Comparison **125** of Collection of Object Representations **525/2** or portions thereof from Object Processing Unit **93** with Collection of Object Representations **525*a*4** or portions thereof from Knowledge Cell **800*oa***. Collection of Object Representations **525*a*4** or portions thereof from Knowledge Cell **800*oa*** may be found substantially similar. Decision-making Unit **540** can anticipate Instruction Sets **526*a*5-526*a*6** correlated with Collection of Object Representations **525*a*4**, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparison **125** of Collection of Object Representations **525/3** or portions thereof from Object Processing Unit **93** with Collection of Object Representations **525*a*5** or portions thereof from Knowledge Cell **800*oa***. Collection of Object Representations **525*a*5** or portions thereof from Knowledge Cell **800*oa*** may be found substantially similar. Decision-making Unit **540** may not anticipate any Instruction Sets **526** since none are correlated with Collection of Object Representations **525*a*5**. Decision-making Unit **540** can implement similar logic or process for any additional Collections of Object Representations **525** from Object Processing Unit **93**, and so on.

It should be understood that any of the described elements and/or techniques in the foregoing example can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells **800** or elements (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof would be affected accordingly. In one example, Extra Info **527** can be included in the Similarity Compari-

sons **125** as previously described. In another example, as history of incoming Collections of Object Representations **525** becomes available, Decision-making Unit **540** can perform collective Similarity Comparisons **125** of the history of Collections of Object Representations **525** or portions thereof from Object Processing Unit **93** with subsequences of Collections of Object Representations **525** or portions thereof from Knowledge Cell **800**. In a further example, traversing may be performed in incremental traversing pattern such as starting from one end of Knowledge Cell **800** and moving the comparison subsequence up or down the list one or any number of incremental Collections of Object Representations **525** at a time. Other traversing patterns or methods can be employed such as starting from the middle of the Knowledge Cell **800** and subdividing the resulting subsequence in a recursive pattern, or any other traversing pattern or method. In a further example, the described traversing of a single Knowledge Cell **800** may be performed on any number of Knowledge Cells **800** sequentially or in parallel. Parallel processors such as a plurality of Processors **11** or cores thereof can be utilized for such parallel processing. In a further example, various arrangements of Collections of Object Representations **525** and/or other elements in a Knowledge Cell **800** can be utilized as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to FIG. **26**, an embodiment of determining anticipatory Instruction Sets **526** using collective similarity comparisons is illustrated. For example, Decision-making Unit **540** can perform Similarity Comparisons **125** of Collection of Object Representations **525/1** or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. Collection of Object Representations **525***c***1** or portions thereof from Knowledge Cell **800***rc* may be found substantially similar with highest similarity. Decision-making Unit **540** can anticipate any Instruction Sets **526** (not shown) correlated with Collection of Object Representations **525***c***1**, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform collective Similarity Comparisons **125** of Collections of Object Representations **525/1**-**525/2** or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. Collections of Object Representations **525***c***1**-**525***c***2** or portions thereof from Knowledge Cell **800***rc* may be found substantially similar with highest similarity. Decision-making Unit **540** can anticipate any Instruction Sets **526** (not shown) correlated with Collection of Object Representations **525***c***2**, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform collective Similarity Comparisons **125** of Collections of Object Representations **525/1**-**525/3** or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. Collections of Object Representations **525***d***1**-**525***d***3** or portions thereof from Knowledge Cell **800***rd* may be found substantially similar with highest similarity. Decision-making Unit **540** can anticipate any Instruction Sets **526** (not shown) correlated with Collection of Object Representations **525***d***3**, thereby enabling autonomous Device **98** operation.

Decision-making Unit **540** can then perform collective Similarity Comparisons **125** of Collections of Object Representations **525/1**-**525/4** or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. Collections of Object Representations **525***d***1**-**525***d***4** or portions thereof from Knowledge Cell **800***rd* may be found substantially similar with highest similarity. Decision-making Unit **540** can anticipate any Instruction Sets **526** (not shown) correlated with Collection of Object Representations **525***d***4**, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform collective Similarity Comparisons **125** of Collections of Object Representations **525/1**-**525/5** or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. Collections of Object Representations **525***d***1**-**525***d***5** or portions thereof from Knowledge Cell **800***rd* may be found substantially similar with highest similarity. Decision-making Unit **540** can anticipate any Instruction Sets **526** (not shown) correlated with Collection of Object Representations **525***d***5**, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can implement similar logic or process for any additional Collections of Object Representations **525** from Object Processing Unit **93**, and so on.

In some embodiments, various elements and/or techniques can be utilized in the aforementioned similarity determinations with respect to collectively compared Collections of Object Representations **525** and/or other elements. In some aspects, similarity of collectively compared Collections of Object Representations **525** can be determined based on similarities or similarity indexes of the individually compared Collections of Object Representations **525**. In one example, an average of similarities or similarity indexes of individually compared Collections of Object Representations **525** can be used to determine similarity of collectively compared Collections of Object Representations **525**. In another example, a weighted average of similarities or similarity indexes of individually compared Collections of Object Representations **525** can be used to determine similarity of collectively compared Collections of Object Representations **525**. For instance, to affect the weighting of collective similarity, a higher weight or importance (i.e. importance index, etc.) can be assigned to the similarities or similarity indexes of some (i.e. more substantive or larger, etc.) Collections of Object Representations **525** and lower for other (i.e. less substantive or smaller, etc.) Collections of Object Representations **525**. Any other higher or lower weight or importance assignment can be implemented. In other aspects, any of the previously described or other thresholds for substantial similarity of individually compared elements can be similarly utilized for collectively compared elements. In one example, substantial similarity of collectively compared Collections of Object Representations **525** can be achieved when their collective similarity or similarity index exceeds a similarity threshold. In another example, substantial similarity of collectively compared Collections of Object Representations **525** can be achieved when at least a threshold number or percentage of Collections of Object Representations **525** or portions thereof from the collectively compared Collections of Object Representations **525** match or substantially match. Similarly, substantial similarity of collectively compared Collections of Object Representations **525** can be achieved when a number or percentage of matching or substantially matching Collec-

tions of Object Representations **525** or portions thereof from the collectively compared Collections of Object Representations **525** exceeds a threshold. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. Similar elements and/or techniques as the aforementioned can be used for similarity determinations of other collectively compared elements such as Object Representations **625**, Object Properties **630**, Instruction Sets **526**, Extra Info **527**, Knowledge Cells **800**, and/or others. Similarity determinations of collectively compared elements may include any features, functionalities, and embodiments of Similarity Comparison **125**, and vice versa.

It should be understood that any of the described elements and/or techniques in the foregoing example can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells **800** or elements (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info **527** in Similarity Comparisons **125**, traversing of Knowledge Cells **800** or other elements, using history of Collections of Object Representations **525** or Knowledge Cells **800** for collective Similarity Comparisons **125**, using various arrangements of Collections of Object Representations **525** and/or other elements in a Knowledge Cell **800**, and/or others can similarly be utilized in this example. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to FIG. **27**, an embodiment of determining anticipatory Instruction Sets **526** using Neural Network **530***a* is illustrated. In some aspects, determining anticipatory Instruction Sets **526** using Neural Network **530***a* may include selecting a path of Knowledge Cells **800** or elements (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof through Neural Network **530***a*. Decision-making Unit **540** can utilize various elements and/or techniques for selecting a path through Neural Network **530***a*. Although, these elements and/or techniques are described with respect to Neural Network **530***a* below, they can similarly be used in any Knowledgebase **530** (i.e. Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.) as applicable.

In some embodiments, Decision-making Unit **540** can utilize similarity index in selecting Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof in a path through Neural Network **530***a*. For instance, similarity index may indicate how well one Knowledge Cell **800** or portions thereof are matched with another Knowledge Cell **800** or portions thereof as previously described. In one example, Decision-making Unit **540** may select a Knowledge Cell **800** comprising one or more Collections of Object Representations **525** with highest similarity index even if Connection **853** pointing to that Knowledge Cell **800** has less than the highest weight. Therefore, similarity index or other such element or parameter can override or disregard the weight of a Connection **853** or other element. In another example, Decision-making Unit **540** may select a Knowledge Cell **800** comprising one or more Collections of Object Representations **525** whose similarity index is higher than or equal to a weight of Connection **853** pointing to that Knowledge Cell **800**. In a further example, Decision-making Unit **540** may select a Knowledge Cell **800** comprising one or more Collections of Object Representations **525** whose similarity index is lower than or equal to a weight of Connection **853** pointing to that Knowledge Cell **800**. Similarity index can be set to be more, less, or equally important than a weight of a Connection **853**.

In some embodiments, Decision-making Unit **540** can utilize Connections **853** in selecting Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof in a path through Neural Network **530***a*. In some aspects, Decision-making Unit **540** can take into account weights of Connections **853** among the interconnected Knowledge Cells **800** in choosing from which Knowledge Cell **800** to compare one or more Collections of Object Representations **525** first, second, third, and so on. Specifically, for instance, Decision-making Unit **540** can perform Similarity Comparisons **125** with one or more Collections of Object Representations **525** from Knowledge Cell **800** pointed to by the highest weight Connection **853** first, Collections of Object Representations **525** from Knowledge Cell **800** pointed to by the second highest weight Connection **853** second, and so on. In other aspects, Decision-making Unit **540** can stop performing Similarity Comparisons **125** as soon as it finds one or more substantially similar Collections of Object Representations **525** in an interconnected Knowledge Cell **800**. In further aspects, Decision-making Unit **540** may only follow the highest weight Connection **853** to arrive at a Knowledge Cell **800** comprising one or more Collections of Object Representations **525** to be compared, thereby disregarding Connections **853** with less than the highest weight. In further aspects, Decision-making Unit **540** may ignore weights and/or other parameters of Connections **853**. In further aspects, Decision-making Unit **540** may ignore Connections **853**.

In some embodiments, Decision-making Unit **540** can utilize a bias to adjust similarity index, weight of a Connection **853**, and/or other element or parameter used in selecting Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof in a path through Neural Network **530***a*. In one example, Decision-making Unit **540** may select a Knowledge Cell **800** comprising one or more Collections of Object Representations **525** whose similarity index multiplied by or adjusted for a bias is higher than or equal to a weight of Connection **853** pointing to that Knowledge Cell **800**. In another example, Decision-making Unit **540** may select a Knowledge Cell **800** comprising one or more Collections of Object Representations **525** whose similarity index multiplied by or adjusted for a bias is lower than or equal to a weight of Connection **853** pointing to that Knowledge Cell **800**. In a further example, bias can be used to resolve deadlock situations where similarity index is equal to a weight of a Connection **853**. In some aspects, bias can be expressed in percentages such as 0.3 percent, 1.2 percent, 25.7 percent, 79.8 percent, 99.9 percent, 100.1 percent, 155.4 percent, 298.6 percent, 1105.5 percent, and so on. For example, a bias below 100 percent decreases an element or parameter to which it is applied, a bias equal to 100 percent does not change the element or parameter to which it is applied, and a bias higher than 100 percent increases the element or parameter to which it is applied. In general, any amount of bias can be utilized depending on implementation. Bias can be applied to one or more of a weight of a Connection **853**, similarity index, any other element or

parameter, and/or all or any combination of them. Also, different biases can be applied to each of a weight of a Connection **853**, similarity index, or any other element or parameter. For example, 30 percent bias can be applied to similarity index and 15 percent bias can be applied to a weight of a Connection **853**. Also, different biases can be applied to various Layers **854** of Neural Network **530***a*, and/or other disclosed elements. Bias can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input.

Any other element and/or technique can be utilized in selecting Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof in a path through Neural Network **530***a*.

In some embodiments, Neural Network **530***a* may include knowledge (i.e. interconnected Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**, etc.) of how Device **98** operated in circumstances including objects with various properties. In some aspects, determining anticipatory Instruction Sets **526** using Neural Network **530***a* may include selecting a path of Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof through Neural Network **530***a*. Individual and/or collective Similarity Comparisons **125** can be used to determine substantial similarity of the individually and/or collectively compared Collections of Object Representations **525** or portions thereof. Substantial similarity may be used primarily for selecting a path through Neural Network **530***a*, whereas, weight of any Connection **853** may be used secondarily or not at all.

For example, Decision-making Unit **540** can perform Similarity Comparisons **125** of Collections of Object Representations **525***a***1**-**525***an* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from one or more Knowledge Cells **800** in Layer **854***a* (or any other one or more Layers **854**, etc.). Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***ta* may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525***b***1**-**525***bn* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from one or more Knowledge Cells **800** in Layer **854***b* interconnected with Knowledge Cell **800***ta*. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***tb* may be found collectively substantially similar with highest similarity, thus, Decision-making Unit **540** may follow Connection **853***t***1** disregarding its less than highest weight. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Since Connection **853***t***2** is the only connection from Knowledge Cell **800***tb*, Decision-making Unit **540** may follow Connection **853***t***2** and perform Similarity Comparisons **125** of Collections of Object Representations **525***c***1**-**525***cn* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***tc* in Layer **854***c*. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***tc* may be found collectively substantially similar. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525***d***1**-**525***dn* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from one or more Knowledge Cells **800** in Layer **854***d* interconnected with Knowledge Cell **800***tc*. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***td* may be found collectively substantially similar with highest similarity, thus, Decision-making Unit **540** may follow Connection **853***t***3**. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525***e***1**-**525***en* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from one or more Knowledge Cells **800** in Layer **854***e* interconnected with Knowledge Cell **800***td*. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***te* may be found collectively substantially similar with highest similarity, thus, Decision-making Unit **540** may follow Connection **853***t***4**. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can implement similar logic or process for any additional Collections of Object Representations **525** from Object Processing Unit **93**, and so on.

The foregoing exemplary embodiment provides an example of utilizing a combination of collective Similarity Comparisons **125**, individual Similarity Comparisons **125**, Connections **853**, and/or other elements or techniques. It should be understood that any of these elements and/or techniques can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof in a path through Neural Network **530***a* would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info **527** in Similarity Comparisons **125**, traversing of Knowledge Cells **800** or other elements, using history of Collections of Object Representations **525** or Knowledge Cells **800** for collective Similarity Comparisons **125**, using various arrangements of Collections of Object Representations **525** and/or other elements in a Knowledge Cell **800**, and/or

others can similarly be utilized in this example. These elements and/or techniques can similarly be utilized in Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, and/or other data structures or arrangements. In some aspects, instead of anticipating Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525**, Decision-making Unit **540** can anticipate instruction Sets **526** correlated with substantially similar streams of Collections of Object Representations **525**. In other aspects, any time that substantial similarity or other similarity threshold is not achieved in compared Collections of Object Representations **525** or portions thereof from any of the Knowledge Cells **800**, Decision-making Unit **540** can decide to look for a substantially or otherwise similar Collections of Object Representations **525** or portions thereof in Knowledge Cells **800** elsewhere in Neural Network **530***a* such as in any Layer **854** subsequent to a current Layer **854**, in the first Layer **854**, in the entire Neural Network **530***a*, and/or others, even if such Knowledge Cell **800** may be unconnected with a prior Knowledge Cell **800**. It should be noted that any of Collections of Object Representations **525***a***1**-**525***an*, Collections of Object Representations **525***b***1**-**525***bn*, Collections of Object Representations **525***c***1**-**525***cn*, Collections of Object Representations **525***d***1**-**525***dn*, Collections of Object Representations **525***e***1**-**525***en*, etc. may include one Collection of Object Representations **525** or a stream of Collections of Object Representations **525**. It should also be noted that any Knowledge Cell **800** may include one Collection of Object Representations **525** or a stream of Collections of Object Representations **525** as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to FIG. **28**, an embodiment of determining anticipatory Instruction Sets **526** using Graph **530***b* is illustrated. Graph **530***b* may include knowledge (i.e. interconnected Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**, etc.) of how Device **98** operated in circumstances including objects with various properties. In some aspects, determining anticipatory Instruction Sets **526** using Graph **530***b* may include selecting a path of Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof through Graph **530***b*. Individual and/or collective Similarity Comparisons **125** can be used to determine substantial similarity of the individually and/or collectively compared Collections of Object Representations **525** or portions thereof. Substantial similarity may be used primarily for selecting a path through Graph **530***b*, whereas, weight of any Connection **853** may be used secondarily or not at all.

For example, Decision-making Unit **540** can perform Similarity Comparisons **125** of Collections of Object Representations **525***a***1**-**525***an* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from one or more Knowledge Cells **800** in Graph **530***b*. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***ua* may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525***b***1**-**525***bn* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from one or more Knowledge Cells **800** in Graph **530***b* interconnected with Knowledge Cell **800***ua* by outgoing Connections **853**. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***ub* may be found collectively substantially similar with highest similarity, thus, Decision-making Unit **540** may follow Connection **853***u***1** disregarding its less than highest weight. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525***c***1**-**525***cn* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from one or more Knowledge Cells **800** in Graph **530***b* interconnected with Knowledge Cell **800***ub* by outgoing Connections **853**. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***uc* may be found collectively substantially similar with highest similarity, thus, Decision-making Unit **540** may follow Connection **853***u***2** disregarding its less than highest weight. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Since Connection **853***u***3** is the only connection from Knowledge Cell **800***uc*, Decision-making Unit **540** may follow Connection **853***u***3** and perform Similarity Comparisons **125** of Collections of Object Representations **525***d***1**-**525***dn* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***ud* in Graph **530***b*. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***ud* may be found collectively substantially similar. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525***e***1**-**525***en* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from one or more Knowledge Cells **800** in Graph **530***b* interconnected with Knowledge Cell **800***ud* by outgoing Connections **853**. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***ue* may be found collectively substantially similar with highest similarity, thus, Decision-making Unit **540** may follow Connection **853***u***4**. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representa-

tions **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can implement similar logic or process for any additional Collections of Object Representations **525** from Object Processing Unit **93**, and so on.

The foregoing exemplary embodiment provides an example of utilizing a combination of collective Similarity Comparisons **125**, individual Similarity Comparisons **125**, Connections **853**, and/or other elements or techniques. It should be understood that any of these elements and/or techniques can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof in a path through Graph **530***b* would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info **527** in Similarity Comparisons **125**, traversing of Knowledge Cells **800** or other elements, using history of Collections of Object Representations **525** or Knowledge Cells **800** in collective Similarity Comparisons **125**, using various arrangements of Collections of Object Representations **525** and/or other elements in a Knowledge Cell **800**, and/or others can similarly be utilized in this example. These elements and/or techniques can similarly be utilized in Neural Network **530***a*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, and/or other data structures or arrangements. In some aspects, instead of anticipating Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525**, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially matching streams of Collections of Object Representations **525**. In other aspects, any time that substantial similarity or other similarity threshold is not achieved in compared Collections of Object Representations **525** or portions thereof of any of the Knowledge Cells **800**, Decision-making Unit **540** can decide to look for a substantially or otherwise similar Collections of Object Representations **525** or portions thereof in Knowledge Cells **800** elsewhere in Graph **530***b* even if such Knowledge Cell **800** may be unconnected with a prior Knowledge Cell **800**. It should be noted that any of Collections of Object Representations **525***a***1**-**525***an*, Collections of Object Representations **525***b***1**-**525***bn*, Collections of Object Representations **525***c***1**-**525***cn*, Collections of Object Representations **525***d***1**-**525***dn*, Collections of Object Representations **525***e***1**-**525***en*, etc. may include one Collection of Object Representations **525** or a stream of Collections of Object Representations **525**. It should also be noted that any Knowledge Cell **800** may include one Collection of Object Representations **525** or a stream of Collections of Object Representations **525** as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to FIG. **29**, an embodiment of determining anticipatory Instruction Sets **526** using Collection of Sequences **530***c* is illustrated. Collection of Sequences **530***c* may include knowledge (i.e. sequences of Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**, etc.) of how Device **98** operated in circumstances including objects with various properties. In some aspects, determining anticipatory Instruction Sets **526** for autonomous Device **98** operation using Collection of Sequences **530***c* may include selecting a Sequence **533** of Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof from Collection of Sequences **530***c*. Individual and/or collective Similarity Comparisons **125** can be used to determine substantial similarity of the individually and/or collectively compared Collections of Object Representations **525** or portions thereof.

For example, Decision-making Unit **540** can perform Similarity Comparisons **125** of Collections of Object Representations **525***a***1**-**525***an* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in one or more Sequences **533** of Collection of Sequences **530***c*. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***ca* in Sequence **533***wc* may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525***a***1**-**525***an* and **525***b***1**-**525***bn* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Sequences **533** of Collection of Sequences **530***c*. Collections of Object Representations **525** or portions thereof from Knowledge Cells **800***ca*-**800***cb* in Sequence **533***wc* may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525***a***1**-**525***an*, **525***b***1**-**525***bn*, and **525***c***1**-**525***cn* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Sequences **533** of Collection of Sequences **530***c*. Collections of Object Representations **525** or portions thereof from Knowledge Cells **800***da*-**800***dc* in Sequence **533***wd* may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525***a***1**-**525***an*, **525***b***1**-**525***bn*, **525***c***1**-**525***cn*, and **525***d***1**-**525***dn* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Sequences **533** of Collection of Sequences **530***c*. Collections of Object Representations **525** or portions thereof from Knowledge Cells **800***da*-**800***dd* in Sequence **533***wd* may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making

Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525*a*1-525*an***, **525*b*1-525*bn***, **525*c*1-525*cn***, **525*d*1-525*dn***, and **525*e*1-525*en*** or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Sequences **533** of Collection of Sequences **530*c***. Collections of Object Representations **525** or portions thereof from Knowledge Cells **800*da*-800*de*** in Sequence **533*wd*** may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can implement similar logic or process for any additional Collections of Object Representations **525** from Object Processing Unit **93**, and so on.

The foregoing exemplary embodiment provides an example of utilizing a combination of collective Similarity Comparisons **125**, individual Similarity Comparisons **125**, and/or other elements or techniques. It should be understood that any of these elements and/or techniques can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Sequence **533** of Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info **527** in Similarity Comparisons **125**, traversing of Knowledge Cells **800** or other elements, using history of Collections of Object Representations **525** or Knowledge Cells **800** in collective Similarity Comparisons **125**, using various arrangements of Collections of Object Representations **525** and/or other elements in a Knowledge Cell **800**, and/or others can similarly be utilized in this example. These elements and/or techniques can similarly be utilized in Neural Network **530*a***, Graph **530*b***, Collection of Knowledge Cells **530*d***, and/or other data structures or arrangements. In some aspects, instead of anticipating Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525**, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially matching streams of Collections of Object Representations **525**. In other aspects, any time that substantial similarity or other similarity threshold is not achieved in compared Collections of Object Representations **525** or portions thereof from any of the Knowledge Cells **800**, Decision-making Unit **540** can decide to look for a substantially or otherwise similar Collections of Object Representations **525** or portions thereof in Knowledge Cells **800** elsewhere in Collection of Sequences **530*c*** such as in different Sequences **533**. It should be noted that any of Collections of Object Representations **525*a*1-525*an***, Collections of Object Representations **525*b*1-525*bn***, Collections of Object Representations **525*c*1-525*cn***, Collections of Object Representations **525*d*1-525*dn***, Collections of Object Representations **525*e*1-525*en***, etc. may include one Collection of Object Representations **525** or a stream of Collections of Object Representations **525**. It should also be noted that any Knowledge Cell **800** may include one Collection of

Object Representations **525** or a stream of Collections of Object Representations **525** as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring now to Modification Interface **130**. Modification Interface **130** comprises the functionality for modifying execution and/or functionality of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. Modification Interface **130** comprises the functionality for modifying execution and/or functionality of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element at runtime. Modification Interface **130** comprises the functionality for modifying execution and/or functionality of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element based on anticipatory Instruction Sets **526**. In one example, Modification Interface **130** comprises the functionality to access, modify, and/or perform other manipulations on runtime engine/environment, virtual machine, operating system, compiler, just-in-time (JIT) compiler, interpreter, translator, execution stack, file, object, data structure, and/or other computing system elements. In another example, Modification Interface **130** comprises the functionality to access, modify, and/or perform other manipulations on memory, storage, bus, interfaces, and/or other computing system elements. In a further example, Modification Interface **130** comprises the functionality to access, modify, and/or perform other manipulations on Processor **11** registers and/or other Processor **11** elements. In a further example, Modification Interface **130** comprises the functionality to access, modify, and/or perform other manipulations on inputs and/or outputs of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. In a further example, Modification Interface **130** comprises the functionality to access, create, delete, modify, and/or perform other manipulations on functions, methods, procedures, routines, subroutines, and/or other elements of Application Program **18**. In a further example, Modification Interface **130** comprises the functionality to access, create, delete, modify, and/or perform other manipulations on source code, bytecode, compiled, interpreted, or otherwise translated code, machine code, and/or other code. In a further example, Modification Interface **130** comprises the functionality to access, create, delete, modify, and/or perform other manipulations on values, variables, parameters, and/or other data or information. Modification Interface **130** comprises any features, functionalities, and embodiments of Acquisition Interface **120**, and vice versa. Modification Interface **130** also comprises other disclosed functionalities.

Modification Interface **130** can employ various techniques for modifying execution and/or functionality of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. In some aspects, some of the previously described techniques and/or tools can be utilized. Code instrumentation, for instance, may involve inserting additional code, overwriting or rewriting existing code, and/or branching to a separate segment of code in Application Program **18** as previously described. For example, instrumented code may include the following:

```
Device1.moveLeft(23);
modifyApplicationQ;
```

In the above sample code, instrumented call to Modification Interface's **130** function (i.e. modifyApplicationQ, etc.) can be placed after a function (i.e. Device1.moveLeft

(**23**), etc.) of Application Program **18**. Similar call to an application modifying function can be placed after or before some or all functions/routines/subroutines, some or all lines of code, some or all statements, some or all instructions or instruction sets, some or all basic blocks, and/or some or all other code segments of Application Program **18**. One or more application modifying function calls can be placed anywhere in Application Program's **18** code and can be executed at any points in Application Program's **18** execution. The application modifying function (i.e. modifyApplication( ), etc.) may include Artificial Intelligence Unit **110**-determined anticipatory Instruction Sets **526** that can modify execution and/or functionality of Application Program **18**. In some embodiments, the previously described obtaining Application Program's **18** instruction sets, data, and/or other information as well as modifying execution and/or functionality of Application Program **18** can be implemented in a single function that performs both tasks (i.e. traceAndModifyApplication( ), etc.).

In some embodiments, various computing systems and/or platforms may provide native tools for modifying execution and/or functionality of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. Independent vendors may provide tools with similar functionalities that can be utilized across different platforms. These tools enable a wide range of techniques or capabilities such as instrumentation, self-modifying code capabilities, dynamic code capabilities, branching, code rewriting, code overwriting, hot swapping, accessing and/or modifying objects or data structures, accessing and/or modifying functions/routines/subroutines, accessing and/or modifying variable or parameter values, accessing and/or modifying processor registers, accessing and/or modifying inputs and/or outputs, providing runtime memory access, and/or other capabilities. One of ordinary skill in art will understand that, while all possible variations of the techniques for modifying execution and/or functionality of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element are too voluminous to describe, these techniques are within the scope of this disclosure.

In one example, modifying execution and/or functionality of Application Program **18** can be implemented through utilizing metaprogramming techniques, which include applications that can self-modify or that can create, modify, and/or manipulate other applications. Self-modifying code, dynamic code, reflection, and/or other techniques can be used to facilitate metaprogramming. In some aspects, metaprogramming is facilitated through a programming language's ability to access and manipulate the internals of the runtime engine directly or via an API. In other aspects, metaprogramming is facilitated through dynamic execution of expressions (i.e. anticipatory Instruction Sets **526**, etc.) that can be created and/or executed at runtime. In yet other aspects, metaprogramming is facilitated through application modification tools, which can perform modifications on an application regardless of whether the application's programming language enables metaprogramming capabilities. Some operating systems may protect an application loaded into memory by restricting access to the loaded application. This protection mechanism can be circumvented by utilizing operating system's, processor's, and/or other low level features or commands to unprotect the loaded application. For example, a self-modifying application may modify the in-memory image of itself. To do so, the application can obtain the in-memory address of its code. The application may then change the operating system's or platform's protection on this memory range allowing it to modify the code (i.e. insert anticipatory Instruction Sets **526**, etc.). In addition to a self-modifying application, one application can utilize similar technique to modify another application. Linux mprotect command or similar commends of other operating systems can be used to change protection (i.e. unprotect, etc.) for a region of memory, for example. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program **18** can be implemented through native capabilities of dynamic, interpreted, and/or scripting programming languages and/or platforms. Most of these languages and/or platforms can perform functionalities at runtime that static programming languages may perform during compilation. Dynamic, interpreted, and/or scripting languages provide native functionalities such as self-modification of code, dynamic code, extending the application, adding new code, extending objects and definitions, and/or other functionalities that can modify an application's execution and/or functionality at runtime. Examples of dynamic, interpreted, and/or scripting languages include Lisp, Perl, PHP, JavaScript, Ruby, Python, Smalltalk, Tcl, VBScript, and/or others. Similar functionalities can also be provided in languages such as Java, C, and/or others using reflection. Reflection includes the ability of an application to examine and modify the structure and behavior of the application at runtime. For example, JavaScript can modify its own code as it runs by utilizing Function object constructor as follows:

```
myFunc=new Function(arg1, arg2, argN, functionBody);
```

The sample code above causes a new function object to be created with the specified arguments and body. The body and/or arguments of the new function object may include new instruction sets (i.e. anticipatory Instruction Sets **526**, etc.). The new function can be invoked as any other function in the original code. In another example, JavaScript can utilize eval method that accepts a string of JavaScript statements (i.e. anticipatory Instruction Sets **526**, etc.) and execute them as if they were within the original code. An example of how eval method can be used to modify an application includes the following JavaScript code:

```
anticipatoryInstr='Device1. moveForward(27);';
if (anticipatoryInstr !=" " && anticipatoryInstr !=null)
{
eval(anticipatoryInstr);
}
```

In the sample code above, Artificial Intelligence Unit **110** may generate anticipatory Instruction Set **526** (i.e. Device1.moveForward(**27**)' for moving a Device1 forward 27 units, etc.) and save it in anticipatoryInstr variable, which eval method can then execute. Lisp is another example of dynamic, interpreted, and/or scripting language that includes similar capabilities as previously described JavaScript. For example, Lisp's compile command can create a function at runtime, eval command may parse and evaluate an expression at runtime, and exec command may execute a given instruction set (i.e. string, etc.) at runtime. In another example, dynamic as well as some non-dynamic languages may provide macros, which combine code introspection and/or eval capabilities. In some aspects, macros can access inner workings of the compiler, interpreter, virtual machine, runtime environment/engine, and/or other components of the computing platform enabling the definition of language-like constructs and/or generation of a complete program or sections thereof. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program **18** can be implemented through dynamic code, dynamic class loading, reflection, and/or other native functionalities of a programming language or platform. In static applications or static programming, a class can be defined and/or loaded at compile time. Conversely, in dynamic applications or dynamic programming, a class can be loaded into a running environment at runtime. For example, Java Runtime Environment (JRE) may not require that all classes be loaded at compile time and class loading can occur when a class is first referenced at runtime. Dynamic class loading enables inclusion or injection of on-demand code and/or functionalities at runtime. System provided or custom class loaders may enable loading of classes into the running environment. Custom class loaders can be created to enable custom functionalities such as, for example, specifying a remote location from which a class can be loaded. In addition to dynamic loading of a pre-defined class, a class can also be created at runtime. In some aspects, a class source code can be created at runtime. A compiler such as javac, com.sun.tools.javac.Main, javax.tools, javax.tools.JavaCompiler, and/or other packages can then be utilized to compile the source code. Javac, com.sun.tools.javac.Main, javax.tools, javax.tools.JavaCompiler, and/or other packages may include an interface to invoke Java compiler from within a running application. A Java compiler may accept source code in a file, string, object (i.e. Java String, StringBuffer, CharSequence, etc.) and/or other source, and may generate Java bytecode (i.e. class file, etc.). Once compiled, a class loader can then load the compiled class into the running environment. In other aspects, a tool such as Javaassist (i.e. Java programming assistant) can be utilized to enable an application to create or modify a class at runtime. Javassist may include a Java library that provides functionalities to create and/or manipulate Java bytecode of an application as well as reflection capabilities. Javassist may provide source-level and bytecode-level APIs. Using the source-level API, a class can be created and/or modified using only source code, which Javassist may compile seamlessly on the fly. Javassist source-level API can therefore be used without knowledge of Java bytecode specification. Bytecode-level API enables creating and/or editing a class bytecode directly. In yet other aspects, similar functionalities to the aforementioned ones may be provided in tools such as Apache Commons BCEL (Byte Code Engineering Library), ObjectWeb ASM, CGLIB (Byte Code Generation Library), and/or others. Once a dynamic code or class is created and loaded, reflection in high-level programming languages such as Java and/or others can be used to manipulate or change the runtime behavior of an application. Examples of reflective programming languages and/or platforms include Java, JavaScript, Smalltalk, Lisp, Python, .NET Common Language Runtime (CLR), Tcl, Ruby, Perl, PHP, Scheme, PL/SQL, and/or others. Reflection can be used in an application to access, examine, modify, and/or manipulate a loaded class and/or its elements. Reflection in Java can be implemented by utilizing a reflection API such as java.lang.Reflect package. The reflection API provides functionalities such as, for example, loading or reloading a class, instantiating a new instance of a class, determining class and instance methods, invoking class and instance methods, accessing and manipulating a class, fields, methods and constructors, determining the modifiers for fields, methods, classes, and interfaces, and/or other functionalities. The above described dynamic code, dynamic class loading, reflection, and/or other functionalities are similarly provided in the .NET platform through its tools such as, for example, System.CodeDom.Compiler namespace, System.Reflection.Emit namespace, and/or other native or other .NET tools. Other platforms in addition to Java and .NET may provide similar tools and/or functionalities. In some designs, dynamic code, dynamic class loading, reflection, and/or other functionalities can be used to facilitate modification of an application by inserting or injecting instruction sets (i.e. anticipatory Instruction Sets **526**, etc.) into a running application. For example, an existing or dynamically created class comprising DCADO Unit **100** functionalities can be loaded into a running application through manual, automatic, or dynamic instrumentation. Once the class is created and loaded, an instance of DCADO Unit **100** class may be constructed. The instance of DCADO Unit **100** can then take or exert control of the application and/or implement alternate instruction sets (i.e. anticipatory Instruction Sets **526**, etc.) at any point in the application's execution. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program **18** can be implemented through independent tools that can be utilized across different platforms. Such tools provide instrumentation and/or other capabilities on more than one platform or computing system and may facilitate application modification or insertion of instruction sets (i.e. anticipatory Instruction Sets **526**, etc.). Examples of these tools include Pin, DynamoRIO, DynInst, Kprobes, KernInst, OpenPAT, DTrace, SystemTap, and/or others. In some aspects, Pin and/or any of its elements, methods, and/or techniques can be utilized for dynamic instrumentation. Pin can perform instrumentation by taking control of an application after it loads into memory. Pin may insert itself into the address space of an executing application enabling it to take control. Pin JIT compiler can then compile and implement alternate code (i.e. anticipatory Instruction Sets **526**, etc.). Pin provides an extensive API for instrumentation at several abstraction levels. Pin supports two modes of instrumentation, JIT mode and probe mode. JIT mode uses a just-in-time compiler to insert instrumentation and recompile program code while probe mode uses code trampolines for instrumentation. Pin was designed for architecture and operating system independence. In other aspects, KernInst and/or any of its elements, methods, and/or techniques can be utilized for dynamic instrumentation. KernInst includes an instrumentation framework designed for dynamically inserting code into a running kernel of an operating system. KernInst implements probe-based dynamic instrumentation where code can be inserted, changed, and/or removed at will. Kerninst API enables client tools to construct their own tools for dynamic kernel instrumentation to suit variety of purposes such as insertion of alternate instruction sets (i.e. anticipatory Instruction Sets **526**, etc.). Client tools can communicate with KernInst over a network (i.e. internet, wireless network, LAW, WAN, etc). Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program **18** can be implemented through utilizing operating system's native tools or capabilities such as Unix ptrace command. Ptrace includes a system call that may enable one process to control another allowing the controller to inspect and manipulate the internal state of its target. Ptrace can be used to modify a running application such as modify an application with alternate instruction sets (i.e. anticipatory Instruction Sets **526**, etc.).

By attaching to an application using the ptrace call, the controlling application can gain extensive control over the operation of its target. This may include manipulation of its instruction sets, execution path, file descriptors, memory, registers, and/or other components. Ptrace can single-step through the target's code, observe and intercept system calls and their results, manipulate the target's signal handlers, receive and send signals on the target's behalf, and/or perform other operations within the target application. Ptrace's ability to write into the target application's memory space enables the controller to modify the running code of the target application. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program **18** can be implemented through utilizing just-in-time (JIT) compiling. JIT compilation (also known as dynamic translation, dynamic compilation, etc.) includes compilation performed during an application's execution (i.e. runtime, etc.). A code can be compiled when it is about to be executed, and it may be cached and reused later without the need for additional compilation. In some aspects, a JIT compiler can convert source code or byte code into machine code. In other aspects, a JIT compiler can convert source code into byte code. JIT compiling may be performed directly in memory. For example, JIT compiler can output machine code directly into memory and immediately execute it. Platforms such as Java, .NET, and/or others may implement JIT compilation as their native functionality. Platform independent tools for custom system design may include JIT compilation functionalities as well. In some aspects, JIT compilation includes redirecting application's execution to a JIT compiler from a specific entry point. For example, Pin can insert its JIT compiler into the address space of an application. Once execution is redirected to it, JIT compiler may receive alternate instruction sets (i.e. anticipatory Instruction Sets **526**, etc.) immediately before their compilation. The JIT compiled instruction sets can be stored in memory or another repository from where they may be retrieved and executed. Alternatively, for example, JIT compiler can create a copy of the original application code or a segment thereof, and insert alternate code (i.e. anticipatory Instruction Sets **526**, etc.) before compiling the modified code copy. In some aspects, JIT compiler may include a specialized memory such as fast cache memory dedicated to JIT compiler functionalities from which the modified code can be fetched rapidly. JIT compilation and/or any compilation in general may include compilation, interpretation, or other translation into machine code, bytecode, and/or other formats or types of code. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program **18** can be implemented through dynamic recompilation. Dynamic recompilation includes recompiling an application or part thereof during execution. An application can be modified with alternate features or instruction sets that may take effect after recompilation. Dynamic recompilation may be practical in various types of applications including object oriented, event driven, forms based, and/or other applications. In a typical windows-based application, most of the action after initial startup occurs in response to user or system events such as moving the mouse, selecting a menu option, typing text, running a scheduled task, making a network connection, and/or other events when an event handler is called to perform an operation appropriate for the event. Generally, when no events are being generated, the application is idle. For example, when an event occurs and an appropriate event handler is called, instrumentation can be implemented in the application's source code to insert alternate instruction sets (i.e. anticipatory Instruction Sets **526**, etc.) at which point the modified source code can be recompiled and/or executed. In some aspects, the state of the application can be saved before recompiling its modified source code so that the application may continue from its prior state. Saving the application's state can be achieved by saving its variables, data structures, objects, location of its current instruction, and/or other necessary information in environmental variables, memory, or other repositories where they can be accessed once the application is recompiled. In other aspects, application's variables, data structures, objects, address of its current instruction, and/or other necessary information can be saved in a repository such as file, database, or other repository accessible to the application after recompilation of its source code. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program **18** can be implemented through modifying or redirecting Application Program's **18** execution path. Generally, an application can be loaded into memory and the flow of execution proceeds from one instruction set to the next until the end of the application. An application may include a branching mechanism that can be driven by keyboard or other input devices, system events, and/or other computing system components or events that may impact the execution path. The execution path can also be altered by an external application through acquiring control of execution and/or redirecting execution to a function, routine/subroutine, or an alternate code segment at any point in the application's execution. A branch, jump, or other mechanism can be utilized to implement the redirected execution. For example, a jump instruction can be inserted at a specific point in an application's execution to redirect execution to an alternate code segment. A jump instruction set may include, for example, an unconditional branch, which always results in branching, or a conditional branch, which may or may not result in branching depending on a condition. When executing an application, a computer may fetch and execute instruction sets in sequence until it encounters a branch instruction set. If the instruction set is an unconditional branch, or it is conditional and the condition is satisfied, the computer may fetch its next instruction set from a different instruction set sequence or code segment as specified by the branch instruction set. After the execution of the alternate code segment, control may be redirected back to the original jump point or to another point in the application. For example, modifying an application can be implemented by redirecting execution of an application to alternate instruction sets (i.e. anticipatory Instruction Sets **526**, etc.). Alternate instruction sets can be pre-compiled, pre-interpreted, or otherwise pre-translated and ready for execution. Alternate instruction sets can also be JIT compiled, JIT interpreted, or otherwise JIT translated before execution. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program **18** can be implemented through assembly language. Assembly language instructions may be directly related with the architecture's machine instructions as previously described. Assembly language

can, therefore, be a powerful tool for implementing direct hardware (i.e. processor registers, memory, etc.) access and manipulations as well as access and manipulations of specialized processor features or instructions. Assembly language can also be a powerful tool for implementing low-level embedded systems, real-time systems, interrupt handlers, self or dynamically modifying code, and/or other applications. Specifically, for instance, self or dynamically modifying code that can be used to facilitate modifying of an application can be seamlessly implemented using assembly language. For example, using assembly language, instruction sets can be dynamically created and loaded into memory similar to the ones that a compiler may generate. Furthermore, using assembly language, memory space of a loaded application can be accessed to modify (including rewrite, overwrite, etc.) original instruction sets or to insert jumps or branches to alternate code elsewhere in memory. Some operating systems may implement protection from changes to applications loaded into memory. Operating system's, processor's, or other low level features or commands can be used to unprotect the protected locations in memory before the change as previously described. Alternatively, a pointer that may reside in a memory location where it could be readily altered can be utilized where the pointer may reference alternate code. In one example, assembly language can be utilized to write alternate code (i.e. anticipatory Instruction Sets **526**, etc.) into a location in memory outside a running application's memory space. Assembly language can then be utilized to redirect the application's execution to the alternate code by inserting a jump or branch into the application's in-memory code, by redirecting program counter, or by other technique. In another example, assembly language can be utilized to overwrite or rewrite the entire or part of an application's in-memory code with alternate code. In some aspects, high-level programming languages can call an external assembly language program to facilitate application modification as previously described. In yet other aspects, relatively low-level programming languages such as C may allow embedding assembly language directly in their source code such as, for example, using asm keyword of C. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program **18** can be implemented through binary rewriting. Binary rewriting tools and/or techniques may modify an application's executable. In some aspects, modification can be minor such as in the case of optimization where the original executable's functionality is kept. In other aspects, modification may change the application's functionality such as by inserting alternate code (i.e. anticipatory Instruction Sets **526**, etc.). Examples of binary rewriting tools include SecondWrite, ATOM, DynamoRIO, Purify, Pin, EEL, DynInst, PLTO, and/or others. Binary rewriting may include disassembly, analysis, and/or modification of target application. Since binary rewriting works directly on machine code executable, it is independent of source language, compiler, virtual machine (if one is utilized), and/or other higher level abstraction layers. Also, binary rewriting tools can perform application modifications without access to original source code. Binary rewriting tools include static rewriters, dynamic rewriters, minimally-invasive rewriters, and/or others. Static binary rewriters can modify an executable when the executable is not in use (i.e. not running). The rewritten executable may then be executed including any new or modified functionality. Dynamic binary rewriters can modify an executable during its execution, thereby enabling modification of an application's functionality at runtime. In some aspects, dynamic rewriters can be used for instrumentation or selective modifications such as insertion of alternate code (i.e. anticipatory Instruction Sets **526**, etc.), and/or for other runtime transformations or modifications. For example, some dynamic rewriters can be configured to intercept an application's execution at indirect control transfers and insert instrumentation or other application modifying code. Minimally-invasive rewriters may keep the original machine code to the greatest extent possible. They support limited modifications such as insertion of jumps into and out of instrumented code. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

Referring to FIG. **30**, in a further example, modifying execution and/or functionality of Processor **11** can be implemented through modification of processor registers, memory, or other computing system components. In some aspects, modifying execution and/or functionality of Processor **11** can be implemented by redirecting Processor's **11** execution to alternate instruction sets (i.e. anticipatory Instruction Sets **526**, etc.). In one example, Program Counter **211** may hold or point to a memory address of the next instruction set that will be executed by Processor **11**. Artificial Intelligence Unit **110** may generate anticipatory Instruction Sets **526** and store them in Memory **12** as previously described. Modification Interface **130** may then change Program Counter **211** to point to the location in Memory **12** where anticipatory Instruction Sets **526** are stored. The anticipatory Instruction Sets **526** can then be fetched from the location in Memory **12** pointed to by the modified Program Counter **211** and loaded into Instruction Register **212** for decoding and execution. Once anticipatory Instruction Sets **526** are executed, Modification Interface **130** may change Program Counter **211** to point to the last instruction set before the redirection or to any other instruction set. In other aspects, anticipatory Instruction Sets **526** can be loaded directly into Instruction Register **212**. As previously described, examples of other processor or computing system components that can be used during an instruction cycle include memory address register (MAR), memory data register (MDR), data registers, address registers, general purpose registers (GPRs), conditional registers, floating point registers (FPRs), constant registers, special purpose registers, machine-specific registers, Register Array **214**, Arithmetic Logic Unit **215**, control unit, and/or other circuits or components. Any of the aforementioned processor registers, memory, or other computing system components can be accessed and/or modified to facilitate the disclosed functionalities. In some embodiments, processor interrupt may be issued to facilitate such access and/or modification. In some designs, modifying execution and/or functionality of Processor **11** can be implemented in a program, combination of programs and hardware, or purely hardware system. Dedicated hardware may be built to perform modifying execution and/or functionality of Processor **11** with marginal or no impact to computing overhead. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

Referring to FIGS. **31**A-**31**B, in a further example, modifying execution and/or functionality of Logic Circuit **250** can be implemented through modification of inputs and/or outputs of Logic Circuit **250**. While Processor **11** includes any type of logic circuit, Logic Circuit **250** is described separately herein to offer additional detail on its functioning. Logic Circuit **250** comprises the functionality for performing logic operations using the circuit's inputs and producing

outputs based on the logic operations performed as previously described. In one example, Logic Circuit **250** may perform some logic operations using four input values and produce two output values. Modifying execution and/or functionality of Logic Circuit **250** can be implemented by replacing its input values with anticipatory input values (i.e. anticipatory Instruction Sets **526**, etc.). Artificial Intelligence Unit **110** may generate anticipatory input values as previously described. Modification Interface **130** can then transmit the anticipatory input values to Logic Circuit **250** through the four hardwired connections as shown in FIG. **31**A. Modification Interface **130** may use Switches **251** to prevent delivery of any input values that may be sent to Logic Circuit **250** from its usual input source. As such, DCADO Unit **100** may cause Logic Circuit **250** to perform its logic operations using the four anticipatory input values, thereby implementing autonomous Device **98** operation. In another example, Logic Circuit **250** may perform some logic operations using four input values and produce two output values. Modifying execution and/or functionality of Logic Circuit **250** can be implemented by replacing its output values with anticipatory output values (i.e. anticipatory Instruction Sets **526**, etc.). Artificial Intelligence Unit **110** may generate anticipatory output values (i.e. anticipatory Instruction Sets **526**, etc.) as previously described. Modification Interface **130** can then transmit the anticipatory output values through the two hardwired connections as shown in FIG. **31**B. Modification Interface **130** may use Switches **251** to prevent delivery of any output values that may be sent by Logic Circuit **250**. As such, DCADO Unit **100** may bypass Logic Circuit **250** and transmit the two anticipatory output values to downstream elements, thereby implementing autonomous Device **98** operation. In a further example, instead of or in addition to modifying input and/or output values of Logic Circuit **250**, the execution and/or functionality of Logic Circuit **250** may be modified by modifying values or signals in one or more Logic Circuit's **250** internal components such as registers, memories, buses, and/or others (i.e. similar to the previously described modifying of Processor **11** components, etc.). In some designs, modifying execution and/or functionality of Logic Circuit **250** can be implemented in a program, combination of programs and hardware, or purely hardware system. Dedicated hardware may be built to perform modifying execution and/or functionality of Logic Circuit **250** with marginal or no impact to computing overhead. Any of the elements and/or techniques for modifying execution and/or functionality of Logic Circuit **250** can similarly be implemented with Processor **11** and/or other processing elements.

In some embodiments, DCADO Unit **100** may directly modify the functionality of an actuator (previously described, not shown). For example, Logic Circuit **250** or other processing element may control an actuator that enables Device **98** to perform mechanical, physical, and/or other operations. An actuator may receive one or more input values or control signals from Logic Circuit **250** or other processing element directing the actuator to perform specific operations. Modifying functionality of an actuator can be implemented by replacing its input values with anticipatory input values (i.e. anticipatory Instruction Sets **526**, etc.) as previously described with respect to replacing input values of Logic Circuit **250**. Specifically, for instance, Artificial Intelligence Unit **110** may generate anticipatory input values as previously described. Modification Interface **130** can then transmit the anticipatory input values to the actuator. Modification Interface **130** may use Switches **251** to prevent delivery of any input values that may be sent to the actuator from its usual input source. As such, DCADO Unit **100** may cause the actuator to perform its operations using the anticipatory input values, thereby implementing autonomous Device **98** operation.

One of ordinary skill in art will recognize that FIGS. **31**A-**31**B depict one of many implementations of Logic Circuit **250** and that any number of input and/or output values can be utilized in alternate implementations. One of ordinary skill in art will also recognize that Logic Circuit **250** may include any number and/or combination of logic components to implement any logic operations.

Other additional techniques or elements can be utilized as needed for modifying execution and/or functionality of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing elements, or some of the disclosed techniques or elements can be excluded, or a combination thereof can be utilized in alternate embodiments.

Referring to FIG. **32**, the illustration shows an embodiment of a method **9100** for learning and/or using a device's circumstances for autonomous device operation. In some aspects, the method can be used on a computing enabled device or system to enable learning of a device's operation in circumstances including objects with various properties and enable autonomous device operation in similar circumstances. Method **9100** may include any action or operation of any of the disclosed methods such as method **9200**, **9300**, **9400**, **9500**, **9600**, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method **9100**.

At step **9105**, a first collection of object representations is received. A collection of object representations (i.e. Collection of Object Representations **525**, etc.) may include one or more object representations (i.e. Object Representations **625**, etc.), object properties (i.e. Object Properties **630**, etc.), and/or other elements or information. An object representation may include an electronic representation of an object (i.e. Object **615**, etc.) detected in a device's surrounding. In some aspects, a collection of object representations may include one or more object representations, object properties, and/or other elements or information detected in a device's (i.e. Device's **98**, etc.) surrounding at a particular time. A collection of object representations may, therefore, include knowledge (i.e. unit of knowledge, etc.) of a device's circumstances including objects with various properties at a particular time. In some designs, a collection of object representations may include or be associated with a time stamp (not shown), order (not shown), or other time related information. In some embodiments, a collection of object representations may include or be substituted with a stream of collections of object representations, and vice versa. Therefore, the terms collection of object representations and stream of collections of object representations may be used interchangeably herein depending on context. A stream of collections of object representations may include one collection of object representations or a group, sequence, or other plurality of collections of object representations. In some aspects, a stream of collections of object representations may include one or more collections of object representations, and/or other elements or information detected in a device's surrounding over time. A stream of collections of object representations may, therefore, include knowledge (i.e. unit of knowledge, etc.) of a device's circumstances including objects with various properties over time. As circumstances including objects with various properties in a device's surrounding change (i.e. objects and/or

their properties change, move, act, transform, etc.) over time, this change may be captured in a stream of collections of object representations. In some designs, each collection of object representations in a stream may include or be associated with the aforementioned time stamp, order, or other time related information. Examples of objects include biological objects (i.e. persons, animals, vegetation, etc.), nature objects (i.e. rocks, bodies of water, etc.), manmade objects (i.e. buildings, streets, ground/aerial/aquatic vehicles, etc.), and/or others. In some aspects, any part of an object may be detected as an object itself. For instance, instead of or in addition to detecting a vehicle as an object, a wheel and/or other parts of the vehicle may be detected as objects. In general, an object may include any object or part thereof that can be detected. Examples of object properties include existence of an object, type of an object (i.e. person, cat, vehicle, building, street, tree, rock, etc.), identity of an object (i.e. name, identifier, etc.), distance of an object, bearing/angle of an object, location of an object (i.e. distance and bearing/angle from a known point, object coordinates, etc.), shape/size of an object (i.e. height, width, depth, computer model, point cloud, etc.), activity of an object (i.e. motion, gestures, etc.), and/or other properties of an object. In general, an object property may include any attribute of an object (i.e. existence of an object, type of an object, identity of an object, shape/size of an object, etc.), any relationship of an object with the device, other objects, or the environment (i.e. distance of an object, bearing/angle of an object, friend/foe relationship, etc.), and/or other information related to an object. Objects and/or their properties can be detected by one or more sensors (i.e. Sensors **92**, etc.) and/or an object processing unit (i.e. Object Processing Unit **93**, etc.). A sensor may obtain or detect information about its environment. As such, one or more sensors can be used to detect objects and/or their properties in a device's surrounding. In some designs, a sensor may be part of a device whose circumstances are being used for DCADO functionalities. In other designs, a sensor may be part of a remote device whose circumstances are being used for DCADO functionalities. Examples of a sensor include a camera (i.e. Camera **92***a*, etc.), a microphone (i.e. Microphone **92***b*, etc.), a lidar (i.e. Lidar **92***c*, etc.), a radar (i.e. Radar **92***d*, etc.), a sonar (i.e. Sonar **92***e*, etc.), and/or others. An object processing unit may process output from a sensor to obtain information of interest. As such, an object processing unit can be used to process output from a sensor to detect objects and/or their properties in a device's surrounding. In some aspects, an object processing unit may create or generate a collection of object representations. In other aspects, an object processing unit may create or generate a stream of collections of object representations. An object processing unit may include a picture recognizer (i.e. Picture Recognizer **94***a*, etc.), a sound recognizer (i.e. Sound Recognizer **94***b*, etc.), a lidar processing unit (i.e. Lidar Processing Unit **94***c*, etc.), a radar processing unit (i.e. Radar Processing Unit **94***d*, etc.), a sonar processing unit (i.e. Sonar Processing Unit **94***e*, etc.), and/or other elements or functionalities. In general, an object processing unit may include any signal processing element or technique known in art as applicable. In some implementations, an object processing unit and/or any of its elements or functionalities can be included in sensor and/or other elements. Receiving comprises any action or operation by or for a Collection of Object Representations **525**, stream of Collections of Object Representations **525**, Object Representation **625**, Object Property **630**, Sensor **92**, Camera **92***a*, Microphone **92***b*, Lidar **92***c*, Radar **92***d*, Sonar **92***e*, Object Processing Unit **93**, Picture Recognizer **94***a*, Sound Recognizer **94***b*, Lidar Processing Unit **94***c*, Radar Processing Unit **94***d*, Sonar Processing Unit **94***e*, and/or other disclosed elements.

At step **9110**, a first one or more instruction sets for operating a device are received. In some embodiments, an instruction set (i.e. Instruction Set **526**, etc.) may be used or executed by a processor (i.e. Processor **11**, etc.) in operating a device. In other embodiments, an instruction set may be part of an application program (i.e. Application Program **18**, etc.) used in operating a device. For example, the application can run or execute on one or more processors or other processing elements. In further embodiments, an instruction set may be used or executed by a logic circuit (i.e. Logic Circuit **250**, etc.) in operating a device. For example, such instruction set may be or include one or more inputs into or outputs from a logic circuit. In further embodiments, an instruction set may be used by an actuator in operating a device. For example, such instruction set may be or include one or more inputs into an actuator. Operating a device includes performing or causing any operations on/by/with the device. In some designs, an instruction set can be received from a processor, application program, logic circuit, and/or other processing element as the instruction set is being used or executed. In other aspects, an instruction set can be received from a processor, application program, logic circuit, and/or other processing element after the instruction set is used or executed. In further aspects, an instruction set can be received from a processor, application program, logic circuit, and/or other processing element before the instruction set has been used or executed. In further aspects, an instruction set can be received from a running processor, running application program, running logic circuit, and/or other running processing element. As such, an instruction set can be received at runtime. In other designs, an instruction set can be received from an actuator. In further designs, an instruction set can be received from memory (i.e. Memory **12**, etc.), hard drive, or any other storage element or repository. In further designs, an instruction set can be received over a network such as Internet, local area network, wireless network, and/or other network. In further designs, an instruction set can be received by an interface (i.e. Acquisition Interface **120**, etc.) configured to obtain instruction sets from a processor, application program, logic circuit, actuator, and/or other element. In general, an instruction set can be received by any element of the system. One or more instruction sets may temporally correspond to a collection of object representations. In some aspects, an instruction set that temporally corresponds to a collection of object representations includes an instruction set used or executed at the time of generating the collection of object representations. In other aspects, an instruction set that temporally corresponds to a collection of object representations includes an instruction set used or executed within a certain time period before and/or after generating the collection of object representations. Any time period can be utilized depending on implementation. In further aspects, an instruction set that temporally corresponds to a collection of object representations includes an instruction set used or executed from the time of generating the collection of object representations to the time of generating a next collection of object representations. In further aspects, an instruction set that temporally corresponds to a collection of object representations includes an instruction set used or executed from the time of generating a preceding collection of object representations to the time of generating the collection of object representations. Any other temporal relationship or correspondence between collections of object representations and correlated

instruction sets can be implemented. In general, one or more instruction sets that temporally correspond to a collection of object representations enable structuring knowledge of a device's operation at or around the time of generating the collection of object representations. Such functionality enables spontaneous or seamless learning of a device's operation in circumstances including objects with various properties as the device is operated in real life situations. In some embodiments, an instruction set may include one or more commands, keywords, symbols (i.e. parentheses, brackets, commas, semicolons, etc.), instructions, operators (i.e. =, <, >, etc.), variables, values, objects, data structures, functions (i.e. Function1( ), FIRST( ), MIN( ), SORT( ), etc.), parameters, states, signals, inputs, outputs, references thereto, and/or other components. In other embodiments, an instruction set may include source code, bytecode, intermediate code, compiled, interpreted, or otherwise translated code, runtime code, assembly code, machine code, and/or any other computer code. In further embodiments, an instruction set may include one or more inputs into and/or outputs from a logic circuit. In further embodiments, an instruction set may include one or more inputs into an actuator. Receiving comprises any action or operation by or for an Acquisition Interface **120**, Instruction Set **526**, and/or other disclosed elements.

At step **9115**, the first collection of object representations is correlated with the first one or more instruction sets for operating the device. In some aspects, individual collections of object representations can be correlated with one or more instruction sets. In other aspects, streams of collections of object representations can be correlated with one or more instruction sets. In further aspects, individual collections of object representations or streams of collections of object representations can be correlated with the aforementioned temporally corresponding instruction sets. In further aspects, a collection of object representations or stream of collections of object representations may not be correlated with any instruction sets. Correlating may include structuring or generating a knowledge cell (i.e. Knowledge Cell **800**, etc.) and storing one or more collections of object representations correlated with any instruction sets into the knowledge cell. Therefore, a knowledge cell may include any data structure or arrangement that can facilitate such storing. A knowledge cell includes knowledge (i.e. unit of knowledge, etc.) of how a device operated in a circumstance including objects with various properties. In some designs, extra information (i.e. Extra Info **527**, etc.) may optionally be used to facilitate enhanced comparisons or decision making in autonomous device operation where applicable. Therefore, any collection of object representations, instruction set, and/or other element may include or be correlated with extra information. Extra information may include any information useful in comparisons or decision making performed in autonomous device operation. Examples of extra information include time information, location information, computed information, contextual information, and/or other information. Correlating may be omitted where learning of a device's operation in circumstances including objects with various properties is not implemented. Correlating comprises any action or operation by or for a Knowledge Structuring Unit **520**, Knowledge Cell **800**, and/or other disclosed elements.

At step **9120**, the first collection of object representations correlated with the first one or more instruction sets for operating the device are stored. A collection of object representations correlated with one or more instruction sets may be part of a stored plurality of collections of object representations correlated with one or more instruction sets.

Collections of object representations correlated with any instruction sets can be stored in a memory unit or other repository. The aforementioned knowledge cells comprising collections of object representations correlated with any instruction sets can be used in/as neurons, nodes, vertices, or other elements in any of the data structures or arrangements (i.e. neural networks, graphs, sequences, collection of knowledge cells, etc.) used for storing the knowledge of a device's operation in circumstances including objects with various properties. Knowledge cells may be connected, interrelated, or interlinked into knowledge structures using statistical, artificial intelligence, machine learning, and/or other models or techniques. Such interconnected or interrelated knowledge cells can be used for enabling autonomous device operation. The interconnected or interrelated knowledge cells may be stored or organized into a knowledgebase (i.e. Knowledgebase **530**, etc.). In some embodiments, knowledgebase may be or include a neural network (i.e. Neural Network **530***a*, etc.). In other embodiments, knowledgebase may be or include a graph (i.e. Graph **530***b*, etc.). In further embodiments, knowledgebase may be or include a collection of sequences (i.e. Collection of Sequences **530***c*, etc.). In further embodiments, knowledgebase may be or include a sequence (i.e. Sequence **533**, etc.). In further embodiments, knowledgebase may be or include a collection of knowledge cells (i.e. Collection of Knowledge Cells **530***d*, etc.). In general, knowledgebase may be or include any data structure or arrangement, and/or repository capable of storing the knowledge of a device's operation in circumstances including objects with various properties. Knowledgebase may also include or be substituted with various artificial intelligence methods, systems, and/or models for knowledge structuring, storing, and/or representation such as deep learning, supervised learning, unsupervised learning, neural networks (i.e. convolutional neural network, recurrent neural network, deep neural network, etc.), search-based, logic and/or fuzzy logic-based, optimization-based, tree/graph/other data structure-based, hierarchical, symbolic and/or sub-symbolic, evolutionary, genetic, multi-agent, deterministic, probabilistic, statistical, and/or other methods, systems, and/or models. Storing may be omitted where learning of a device's operation in circumstances including objects with various properties is not implemented. Storing comprises any action or operation by or for a Knowledgebase **530**, Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, Knowledge Cell **800**, Node **852**, Layer **854**, Connection **853**, Similarity Comparison **125**, and/or other disclosed elements.

At step **9125**, a new collection of object representations is received. Step **9125** may include any action or operation described in Step **9105** as applicable.

At step **9130**, the new collection of object representations is compared with the first collection of object representations. Comparing one collection of object representations with another collection of object representations may include comparing at least a portion of one collection of object representations with at least a portion of the other collection of object representations. In some embodiments, collections of object representations may be compared individually. In some aspects, comparing of individual collections of object representations may include comparing one or more object representations of one collection of object representations with one or more object representations of another collection of object representations. In other aspects, comparing of object representations may include comparing one or more object properties of one object representation

with one or more object properties of another object representation. In some designs, one or more object properties in the same category (i.e. Category **635**, etc.) can be compared. Comparing may include any techniques for comparing text, numbers, and/or other data. In further aspects, some object representations, object properties, and/or other elements of a collection of object representations can be omitted from comparison depending on implementation. In other embodiments, collections of object representations may be compared collectively as part of streams of collections of object representations.

Collective comparing of collections of object representations may include any features, functionalities, and embodiments of the aforementioned individual comparing of collections of object representations. In some aspects, collective comparing of collections of object representations may include comparing one or more collections of object representations of one stream of collections of object representations with one or more collections of object representations of another stream of collections of object representations. In some designs, one or more corresponding (i.e. similarly ordered, temporally related, etc.) collections of object representations from the compared streams of collections of object representations can be compared. In other designs, Dynamic Time Warping (DTW) and/or other techniques can be utilized for comparison and/or aligning temporal sequences (i.e. streams of collections of object representations, etc.) that may vary in time or speed. In further aspects, some collections of object representations can be omitted from comparison depending on implementation. Any combination of the aforementioned and/or other elements or techniques can be utilized in alternate embodiments of the comparing. Comparing may be omitted where anticipating of a device's operation in circumstances including objects with various properties is not implemented. Comparing comprises any action or operation by or for a Decision-making Unit **540**, Similarity Comparison **125**, and/or other disclosed elements.

At step **9135**, a determination is made that there is at least a partial match between the new collection of object representations and the first collection of object representations. In some embodiments, determining at least a partial match between individually compared collections of object representations includes determining that a similarity between one or more portions of one collection of object representations and one or more portions of another collection of object representations exceeds a similarity threshold. In other embodiments, determining at least a partial match between individually compared collections of object representations includes determining at least a partial match between one or more portions of one collection of object representations and one or more portions of another collection of object representations. In further embodiments, determining at least a partial match between individually compared collections of object representations includes determining substantial similarity between one or more portions of one collection of object representations and one or more portions of another collection of object representations. A portion of a collection of object representations may include an object representation, an object property, and/or other portion or element of the collection of object representations. In further embodiments, determining at least a partial match between individually compared collections of object representations includes determining that the number or percentage of matching or substantially matching object representations of the compared collections of object representations exceeds a threshold number (i.e. 1, 2, 4, 7, 18, etc.) or threshold percentage (i.e. 41%, 62%, 79%, 85%, 93%, etc.). In some aspects, type of object representations, importance of object representations, and/or other elements or techniques relating to object representations can be utilized for determining similarity using object representations. In further aspects, some of the object representations can be omitted in determining similarity using object representations depending on implementation. In further embodiments, determining a match or substantial match between compared object representations includes determining that the number or percentage of matching or substantially matching object properties of the compared object representations exceeds a threshold number (i.e. 1, 2, 3, 6, 11, etc.) or a threshold percentage (i.e. 55%, 61%, 78%, 82%, 99%, etc.). In some aspects, categories of object properties, importance of object properties, and/or other elements or techniques relating to object properties can be utilized for determining similarity using object properties. In further aspects, some of the object properties can be omitted in determining similarity using object properties depending on implementation. In some designs, substantial similarity of individually compared collections of object representations can be achieved when a similarity between one or more portions of one collection of object representations and one or more portions of another collection of object representations exceeds a similarity threshold. In other designs, substantial similarity of individually compared collections of object representations can be achieved when the number or percentage of matching or substantially matching object representations of the compared collections of object representations exceeds a threshold number (i.e. 1, 2, 4, 7, 18, etc.) or threshold percentage (i.e. 41%, 62%, 79%, 85%, 93%, etc.). In further aspects, substantial similarity of compared object representations can be achieved when the number or percentage of matching or substantially matching object properties of the compared object representations exceeds a threshold number (i.e. 1, 2, 3, 6, 11, etc.) or a threshold percentage (i.e. 55%, 61%, 78%, 82%, 99%, etc.). In some embodiments, determining at least a partial match between collectively compared collections of object representations (i.e. streams of collections of object representations, etc.) includes determining that the number or percentage of matching or substantially matching collections of object representations of the compared streams of collections of object representations exceeds a threshold number (i.e. 1, 2, 4, 9, 33, 138, etc.) or threshold percentage (i.e. 39%, 58%, 77%, 88%, 94%, etc.). In some aspects, importance of collections of object representations, order of collections of object representations, and/or other elements or techniques relating to collections of object representations can be utilized for determining similarity of collectively compared collections of object representations or streams of collections of object representations. In further aspects, some of the collections of object representations can be omitted in determining similarity of collectively compared collections of object representations or streams of collections of object representations depending on implementation. In some designs, a threshold for a number or percentage similarity can be used to determine a match or substantial match between any of the aforementioned elements. Any text, number, and/or other data similarity determination techniques can be used in any of the aforementioned similarity determinations. A partial match of any of the compared elements may include a substantially or otherwise similar match, and vice versa. Therefore, these terms may be used interchangeably herein depending on context. Although, substantial similarity or substantial match is frequently used

herein, it should be understood that any level of similarity, however high or low, may be utilized as defined by the rules (i.e. thresholds, etc.) for similarity. Any combination of the aforementioned and/or other elements or techniques can be utilized in alternate embodiments. Determining may be omitted where anticipating of a device's operation in circumstances including objects with various properties is not implemented. Determining comprises any action or operation by or for a Decision-making Unit **540**, Similarity Comparison **125**, and/or other disclosed elements.

At step **9140**, the first one or more instruction sets for operating the device correlated with the first collection of object representations are executed. Executing may be performed in response to the aforementioned determining. Executing may be caused by DCADO Unit **100**, Artificial Intelligence Unit **110**, Modification Interface **130**, and/or other disclosed elements. An instruction set may be executed by a processor (i.e. Processor **11**, etc.), application program (i.e. Application Program **18**, etc.), logic circuit (i.e. Logic Circuit **250**, etc.), and/or other processing element. An instruction set may be executed or acted upon by an actuator. In some aspects, instruction sets (i.e. the one or more instruction sets for operating the device correlated with the first collection of object representations, etc.) anticipated or determined to be used or executed in a device's autonomous operation may be referred to as anticipatory instruction sets, alternate instruction sets, and/or other suitable name or reference. Therefore, these terms can be used interchangeably herein depending on context. Executing may include executing one or more alternate instruction sets (i.e. anticipatory instruction sets, etc.) instead of or prior to an instruction set that would have been executed in a regular course of execution. In some embodiments, executing may include modifying a register or other element of a processor with one or more alternate instruction sets. Executing may also include redirecting a processor to one or more alternate instruction sets. In other embodiments, processor may be or comprises a logic circuit. Executing may further include modifying an element of a logic circuit with one or more alternate instruction sets, redirecting the logic circuit to one or more alternate instruction sets, replacing the inputs into the logic circuit with one or more alternate inputs or instruction sets, and/or replacing the outputs from the logic circuit with one or more alternate outputs or instruction sets. Executing may further include replacing the inputs into an actuator with one or more alternate inputs or instruction sets. In further embodiments, a processor may run an application including instruction sets for operating a device. In some aspects, executing includes executing one or more alternate instruction sets as part of the application. In other aspects, executing includes modifying the application. In further aspects, executing includes redirecting the application to one or more alternate instruction sets. In further aspects, executing includes modifying one or more instruction sets of the application. In further aspects, executing includes modifying the application's source code, bytecode, intermediate code, compiled code, interpreted code, translated code, runtime code, assembly code, machine code, or other code. In further aspects, executing includes modifying memory, processor register, storage, repository or other element where the application's instruction sets are stored or used. In further aspects, executing includes modifying instruction sets used for operating an object of the application. In further aspects, executing includes modifying an element of a processor, an element of a device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input used in running the application. In further aspects, executing includes modifying the application at source code write time, compile time, interpretation time, translation time, linking time, loading time, runtime, or other time. In further aspects, executing includes modifying one or more of the application's lines of code, statements, instructions, functions, routines, subroutines, basic blocks, or other code segments. In further aspects, executing includes a manual, automatic, dynamic, just in time (JIT), or other instrumentation of the application. In further aspects, executing includes utilizing one or more of a .NET tool, .NET application programming interface (API), Java tool, Java API, operating system tool, independent tool, or other tool for modifying the application. In further aspects, executing includes utilizing a dynamic, interpreted, scripting, or other programming language. In further aspects, executing includes utilizing dynamic code, dynamic class loading, or reflection. In further aspects, executing includes utilizing assembly language. In further aspects, executing includes utilizing metaprogramming, self-modifying code, or an application modification tool. In further aspects, executing includes utilizing just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further aspects, executing includes utilizing dynamic expression creation, dynamic expression execution, dynamic function creation, or dynamic function execution. In further aspects, executing includes adding or inserting additional code into the application's code. In further aspects, executing includes modifying, removing, rewriting, or overwriting the application's code. In further aspects, executing includes branching, redirecting, extending, or hot swapping the application's code. Branching or redirecting an application's code may include inserting a branch, jump, or other means for redirecting the application's execution. Executing comprises any action or operation by or for a Processor **11**, Application Program **18**, Logic Circuit **250**, Modification Interface **130**, and/or other disclosed elements.

At step **9145**, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations are performed by the device. The one or more operations may be performed in response to the aforementioned executing. In some aspects, an operation includes any operation that can be performed by/with/on a computing enabled device. In other aspects, an operation includes any operation that can be performed by/with/on an actuator. In further aspects, an operation includes any operation that can be performed by/with/on a computer. In general, an operation includes any operation that can be performed by/with/on a device or element thereof. One of ordinary skill in art will recognize that, while all possible variations of operations by/with/on a device are too voluminous to describe and limited only by the device's design and/or user's utilization, all operations are within the scope of this disclosure in various implementations.

Referring to FIG. **33**, the illustration shows an embodiment of a method **9200** for learning and/or using a device's circumstances for autonomous device operation. In some aspects, the method can be used on a computing enabled device or system to enable learning of a device's operation in circumstances including objects with various properties and enable autonomous device operation in similar circumstances. Method **9200** may include any action or operation of any of the disclosed methods such as method **9100**, **9300**, **9400**, **9500**, **9600**, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different

combination or order thereof can be implemented in alternate embodiments of method **9200**.

At step **9205**, a first collection of object representations is received. Step **9205** may include any action or operation described in Step **9105** of method **9100** as applicable.

At step **9210**, a first one or more instruction sets for operating a device are received. Step **9210** may include any action or operation described in Step **9110** of method **9100** as applicable.

At step **9215**, the first collection of object representations correlated with the first one or more instruction sets for operating the device are learned. Step **9215** may include any action or operation described in Step **9115** and/or Step **9120** of method **9100** as applicable.

At step **9220**, a new collection of object representations is received. Step **9220** may include any action or operation described in Step **9125** of method **9100** as applicable.

At step **9225**, the first one or more instruction sets for operating the device correlated with the first collection of object representations are anticipated based on at least a partial match between the new collection of object representations and the first collection of object representations. Step **9225** may include any action or operation described in Step **9130** and/or Step **9135** of method **9100** as applicable.

At step **9230**, the first one or more instruction sets for operating the device correlated with the first collection of object representations are executed. Step **9230** may include any action or operation described in Step **9140** of method **9100** as applicable.

At step **9235**, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations are performed by the device. Step **9235** may include any action or operation described in Step **9145** of method **9100** as applicable.

Referring to FIG. **34**, the illustration shows an embodiment of a method **9300** for learning and/or using a device's circumstances for autonomous device operation. In some aspects, the method can be used on a computing enabled device or system to enable learning of a device's operation in circumstances including objects with various properties and enable autonomous device operation in similar circumstances. Method **9300** may include any action or operation of any of the disclosed methods such as method **9100**, **9200**, **9400**, **9500**, **9600**, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method **9300**.

At step **9305**, a first stream of collections of object representations is received. Step **9305** may include any action or operation described in Step **9105** of method **9100** as applicable.

At step **9310**, a first one or more instruction sets for operating a device are received. Step **9310** may include any action or operation described in Step **9110** of method **9100** as applicable.

At step **9315**, the first stream of collections of object representations is correlated with the first one or more instruction sets for operating the device. Step **9315** may include any action or operation described in Step **9115** of method **9100** as applicable.

At step **9320**, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device are stored. Step **9320** may include any action or operation described in Step **9120** of method **9100** as applicable.

At step **9325**, a new stream of collections of object representations is received. Step **9325** may include any action or operation described in Step **9125** of method **9100** as applicable.

At step **9330**, the new stream of collections of object representations is compared with the first stream of collections of object representations. Step **9330** may include any action or operation described in Step **9130** of method **9100** as applicable.

At step **9335**, a determination is made that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. Step **9335** may include any action or operation described in Step **9135** of method **9100** as applicable.

At step **9340**, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations are executed. Step **9340** may include any action or operation described in Step **9140** of method **9100** as applicable.

At step **9345**, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations are performed by the device. Step **9345** may include any action or operation described in Step **9145** of method **9100** as applicable.

Referring to FIG. **35**, the illustration shows an embodiment of a method **9400** for learning and/or using a device's circumstances for autonomous device operation. In some aspects, the method can be used on a computing enabled device or system to enable learning of a device's operation in circumstances including objects with various properties and enable autonomous device operation in similar circumstances. Method **9400** may include any action or operation of any of the disclosed methods such as method **9100**, **9200**, **9300**, **9500**, **9600**, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method **9400**.

At step **9405**, a first collection of object representations is received. Step **9405** may include any action or operation described in Step **9105** of method **9100** as applicable.

At step **9410**, a first one or more inputs are received, wherein the first one or more inputs are also received by a logic circuit, and wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device. Step **9410** may include any action or operation described in Step **9110** of method **9100** as applicable.

At step **9415**, the first collection of object representations is correlated with the first one or more inputs. Step **9415** may include any action or operation described in Step **9115** of method **9100** as applicable.

At step **9420**, the first collection of object representations correlated with the first one or more inputs are stored. Step **9420** may include any action or operation described in Step **9120** of method **9100** as applicable.

At step **9425**, a new collection of object representations is received. Step **9425** may include any action or operation described in Step **9125** of method **9100** as applicable.

At step **9430**, the new collection of object representations is compared with the first collection of object representations. Step **9430** may include any action or operation described in Step **9130** of method **9100** as applicable.

At step **9435**, a determination is made that there is at least a partial match between the new collection of object repre-

sentations and the first collection of object representations. Step **9435** may include any action or operation described in Step **9135** of method **9100** as applicable.

At step **9440**, the first one or more inputs correlated with the first collection of object representations are received by the logic circuit. Step **9440** may include any action or operation described in Step **9140** of method **9100** as applicable.

At step **9445**, one or more operations defined by one or more outputs for operating the device produced by the logic circuit are performed by the device. Step **9445** may include any action or operation described in Step **9145** of method **9100** as applicable.

Referring to FIG. **36**, the illustration shows an embodiment of a method **9500** for learning and/or using a device's circumstances for autonomous device operation. In some aspects, the method can be used on a computing enabled device or system to enable learning of a device's operation in circumstances including objects with various properties and enable autonomous device operation in similar circumstances. Method **9500** may include any action or operation of any of the disclosed methods such as method **9100**, **9200**, **9300**, **9400**, **9600**, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method **9500**.

At step **9505**, a first collection of object representations is received. Step **9505** may include any action or operation described in Step **9105** of method **9100** as applicable.

At step **9510**, a first one or more outputs are received, the first one or more outputs transmitted from a logic circuit, wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device. Step **9510** may include any action or operation described in Step **9110** of method **9100** as applicable.

At step **9515**, the first collection of object representations is correlated with the first one or more outputs. Step **9515** may include any action or operation described in Step **9115** of method **9100** as applicable.

At step **9520**, the first collection of object representations correlated with the first one or more outputs are stored. Step **9520** may include any action or operation described in Step **9120** of method **9100** as applicable.

At step **9525**, a new collection of object representations is received. Step **9525** may include any action or operation described in Step **9125** of method **9100** as applicable.

At step **9530**, the new collection of object representations is compared with the first collection of object representations. Step **9530** may include any action or operation described in Step **9130** of method **9100** as applicable.

At step **9535**, a determination is made that there is at least a partial match between the new collection of object representations and the first collection of object representations. Step **9535** may include any action or operation described in Step **9135** of method **9100** as applicable.

At step **9540**, one or more operations defined by the first one or more outputs correlated with the first collection of object representations are performed by the device. Step **9540** may include any action or operation described in Step **9145** of method **9100** as applicable.

Referring to FIG. **37**, the illustration shows an embodiment of a method **9600** for learning and/or using a device's circumstances for autonomous device operation. In some aspects, the method can be used on a computing enabled device or system to enable learning of a device's operation in circumstances including objects with various properties and enable autonomous device operation in similar circumstances. Method **9600** may include any action or operation of any of the disclosed methods such as method **9100**, **9200**, **9300**, **9400**, **9500**, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method **9600**.

At step **9605**, a first collection of object representations is received. Step **9605** may include any action or operation described in Step **9105** of method **9100** as applicable.

At step **9610**, a first one or more inputs are received, wherein the first one or more inputs are also received by an actuator, and wherein the actuator is configured to receive inputs and perform motions. Step **9610** may include any action or operation described in Step **9110** of method **9100** as applicable.

At step **9615**, the first collection of object representations is correlated with the first one or more inputs. Step **9615** may include any action or operation described in Step **9115** of method **9100** as applicable.

At step **9620**, the first collection of object representations correlated with the first one or more inputs are stored. Step **9620** may include any action or operation described in Step **9120** of method **9100** as applicable.

At step **9625**, a new collection of object representations is received. Step **9625** may include any action or operation described in Step **9125** of method **9100** as applicable.

At step **9630**, the new collection of object representations is compared with the first collection of object representations. Step **9630** may include any action or operation described in Step **9130** of method **9100** as applicable.

At step **9635**, a determination is made that there is at least a partial match between the new collection of object representations and the first collection of object representations. Step **9635** may include any action or operation described in Step **9135** of method **9100** as applicable.

At step **9640**, the first one or more inputs correlated with the first collection of object representations are received by the actuator. Step **9640** may include any action or operation described in Step **9140** of method **9100** as applicable.

At step **9645**, one or more motions defined by the first one or more inputs correlated with the first collection of object representations are performed by the actuator. Step **9645** may include any action or operation described in Step **9145** of method **9100** as applicable.

Referring to FIG. **38**, in some exemplary embodiments, Device **98** may be or include Loader **98**_a_. Loader **98**_a_ may be operated by User **50** in person or remotely. Loader **98**_a_ may include or be coupled to one or more Sensors **92** (i.e. collectively referred to as Sensor **92**, etc.) such as Camera **92**_a_, Microphone **92**_b_, Lidar **92**_c_, Radar **92**_d_, Sonar **92**_e_, etc. and/or Object Processing Unit **93** that can detect Objects **615**_aa_-**615**_ad_, and/or other elements or information in Loader's **98**_a_ surrounding. Object Processing Unit **93** may include Picture Recognizer **94**_a_, Sound Recognizer **94**_b_, Lidar Processing Unit **94**_c_, Radar Processing Unit **94**_d_, Sonar Processing Unit **94**_e_, and/or other elements or functionalities as applicable. Object Processing Unit **93** may create or generate one or more (i.e. stream, etc.) Collections of Object Representations **525** comprising Object Representations **625**, Object Properties **630**, and/or other elements or information representing Objects **615** detected in Loader's **98**_a_ surrounding. Loader **98**_a_ may also include or be controlled by Logic Circuit **250** (i.e. microcontroller, etc.), Processor **11** (i.e. including any Application Program **18**

running thereon, etc.), and/or other processing element that receives User's **50** (i.e. operator's, etc.) operating directions and causes desired operations with Loader **98***a* such as moving, maneuvering, collecting, lifting, unloading, and/or others. User **50** can interact with Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element through inputting operating directions via Human-machine Interface **23** such as one or more steering wheels, levers, pedals, buttons, or other input devices. For instance, responsive to User's **50** manipulating a steering wheel and one or more levers, Logic Circuit **250** or Processor **11** may cause Loader's **98***a* arm with bucket to collect a load, one or more motors or other actuators to move or maneuver Loader **98***a*, lifting system (i.e. hydraulic, pneumatic, mechanical, electrical, etc.) to lift a load, and/or arm with bucket to unload a load. Loader **98***a* may also include or be coupled to DCADO Unit **100**. DCADO Unit **100** may be embedded (i.e. integrated, etc.) into or coupled to Loader's **98***a* Logic Circuit **250**, Processor **11**, and/or other processing element. DCADO Unit **100** may also be a program embedded (i.e. integrated, etc.) into or interfaced with Application Program **18** running on Processor **11** and/or other processing element. DCADO Unit **100** can obtain Instruction Sets **526** used or executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element. In some aspects, Instruction Sets **526** may include one or more inputs into or outputs from Loader's **98***a* Logic Circuit **250** (i.e. microcontroller, etc.). In other aspects, Instruction Sets **526** may include one or more instruction sets from Loader's **98***a* Processor's **11** registers or other components. In further aspects, Instruction Sets **526** may include one or more instruction sets used or executed in Application Program **18**. DCADO Unit **100** may also optionally obtain any Extra Info **527** (i.e. time, location, computed, contextual, and/or other information, etc.) related to Loader's **98***a* operation. As User **50** operates Loader **98***a* in circumstances including objects with various properties as shown, DCADO Unit **100** may learn Loader's **98***a* operations in these circumstances by correlating Collections of Object Representations **525** representing Objects **615** detected in Loader's **98***a* surrounding with one or more Instruction Sets **526** used or executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element. Any Extra Info **527** related to Loader's **98***a* operation may also optionally be correlated with Collections of Object Representations **525**. DCADO Unit **100** may store this knowledge into Knowledgebase **530** (i.e. Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.). In the future, DCADO Unit **110** may compare incoming Collections of Object Representations **525** representing Objects **615** detected in Loader's **98***a* surrounding with previously learned Collections of Object Representations **525** including optionally using any Extra Info **527** for enhanced decision making. If substantially similar or at least a partial match is found or determined, the Instruction Sets **526** correlated with the previously learned Collections of Object Representations **525** can be autonomously executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element, thereby enabling autonomous operation of Loader **98***a* in similar circumstances as in previously learned ones. For instance, Loader **98***a* comprising DCADO Unit **100** may learn User **50**-directed collecting, moving, maneuvering, lifting, unloading, and/or other operations in a circumstance that includes Rock **615***aa*, Pile of Material **615***ab*, Person **615***ac*, Truck **615***ad*, and/or other Objects **615** among which Loader **98***a* may need to maneuver and/or with which Loader **98***a* may need to interact. In the future, when a circumstance that includes Objects **615** with similar Object Properties **630** is encountered, Loader **98***a* may implement collecting, moving, maneuvering, lifting, and/or unloading operations autonomously.

In some embodiments, DCADO Unit **100** may reside on Server **96** accessible over Network **95** as previously described. In such embodiments, any number of Loaders **98***a* may connect to such remote DCADO Unit **100** and the remote DCADO Unit **100** may learn their operations in circumstances including objects with various properties. In turn, any number of Loaders **98***a* can utilize the remote DCADO Unit **100** for autonomous operation in circumstances including objects with various properties. For example, multiple operators (i.e. Users **50**, etc.) may operate their Loaders **98***a* that are configured to transmit their operations in circumstances including objects with various properties to a remote DCADO Unit **100**. Such remote DCADO Unit **100** enables learning of the operators' collective knowledge of operating Loaders **98** in circumstances including objects with various properties. Any number of Loaders **98** can utilize such collective knowledge comprised in the remote DCADO Unit **100** for their autonomous operation. Any of the disclosed elements such as Artificial Intelligence Unit **110**, Knowledgebase **530**, and/or others may reside on Server **96**, and any combination of local and remote elements can be implemented in alternate embodiments.

In some embodiments, Loader **98***a* may include or be coupled to a plurality of Sensors **92** and/or their corresponding Object Processing Units **93**. In one example, multiple Sensors **92** may detect objects and/or their properties from different angles or on different sides of Loader **98***a*. In another example, one or more Sensors **92** may be placed on different sub-devices, sub-systems, or elements of Loader **98***a*. For instance, one Sensor **92** may be placed on the roof of Loader **98***a*, another Sensor **92** may be placed on the arm of Loader **98***a*, and an additional Sensor **92** may be placed on the bucket of Loader **98***a*. In some designs where multiple Sensors **92** are placed on different sub-devices, sub-systems, or elements of Loader **98***a*, multiple DCADO Units **100** can be utilized (i.e. one DCADO Unit **100** for each Sensor **92** or group of Sensors **92** and/or their corresponding Object Processing Units **93**, etc.). In such designs, as User **50** operates Loader **98***a* in circumstances including objects with various properties, a particular DCADO Unit **100** may learn operations of Loader's **98***a* sub-device, sub-system, or element in these circumstances by correlating Collections of Object Representations **525** representing Objects **615** detected by Sensor **92** on the sub-device, sub-system, or element assigned to the DCADO Unit **100** with one or more Instruction Sets **526** used or executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element. The learning and/or decision making in Loader's **98***a* operation can, therefore, be performed per individual sub-device, sub-system, or element. In other designs where multiple Sensors **92** are placed on different sub-devices, sub-systems, or elements of Loader **98***a*, as User **50** operates Loader **98***a* in circumstances including objects with various properties, a single DCADO Unit **100** may learn Loader's **98***a* operations in these circumstances by correlating collective Collections of Object Representations **525** representing Objects **615** detected by Sensors **92** on the sub-devices, sub-systems, or elements with one or more Instruction Sets **526** used or executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element.

In some embodiments, Loader **98***a* may include a plurality of Logic Circuits **250**, Processors **11**, Application Programs **18**, and/or other processing elements. In some aspects, one or more sub-devices, sub-systems, or elements of Loader **98***a* may be controlled by different processing elements. For example, one Processor **11** (i.e. including any Application Programs **18** running thereon, etc.) may control the moving system (i.e. drivetrain, powertrain, etc.) of Loader **98***a*, one Logic Circuit **250** may control an arm of Loader **98***a*, and an additional Logic Circuit **250** may control a bucket of Loader **98***a*. In some designs where multiple processing elements are utilized, multiple DCADO Units **100** can also be utilized (i.e. one DCADO Unit **100** for each processing element, etc.). In such designs, as User **50** operates Loader **98***a* in circumstances including objects with various properties, a particular DCADO Unit **100** may learn Loader's **98***a* operations in these circumstances by correlating Collections of Object Representations **525** with one or more Instruction Sets **526** used or executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element assigned to the DCADO Unit **100**. The learning and/or decision making in Loader's **98***a* operation can, therefore, be performed per individual processing element. In other designs where multiple processing elements are utilized, as User **50** operates Loader **98***a* in circumstances including objects with various properties, a single DCADO Unit **100** may learn Loader's **98***a* operations in these circumstances by correlating Collections of Object Representations **525** with collective Instruction Sets **526** used or executed by a plurality of Logic Circuits **250**, Processors **11**, Application Programs **18**, and/or other processing elements.

In some embodiments, a combination of DCADO Unit **100** and other systems and/or techniques can be utilized to implement Loader's **98***a* operation. In one example, DCADO Unit **100** may be a primary or preferred system for implementing Loader's **98***a* operation. While operating autonomously under the control of DCADO Unit **100**, Loader **98***a* may encounter a circumstance including objects with various properties that has not been encountered or learned before. In such situations, User **50** and/or non-DCADO system may take control of Loader's **98***a* operation. DCADO Unit **100** may take control again when Loader **98***a* encounters a previously learned circumstance including objects with various properties. Naturally, DCADO Unit **100** can learn Loader's **98***a* operation in the circumstances while User **50** and/or non-DCADO system is in control of Loader **98***a*, thereby reducing or eliminating the need for future involvement of User **50** and/or non-DCADO system. For instance, one User **50** can control or assist in controlling multiple Loaders **98***a* comprising DCADO Units **100**. In such instances, User **50** can control or assist in controlling a Loader **98***a* that may encounter a circumstance including objects with various properties that has not been encountered or learned before while the Loaders **98***a* operating in previously learned circumstances can operate autonomously. In another example, User **50** and/or non-DCADO system may be a primary or preferred system for implementing Loader's **98***a* operation. While operating under the control of User **50** and/or non-DCADO system, User **50** and/or non-DCADO system may release control to DCADO Unit **100** for any reason (i.e. User **50** gets tired or distracted, non-DCADO system gets stuck or cannot make a decision, etc.), at which point Loader **98***a* can be controlled by DCADO Unit **100**. In some designs, DCADO Unit **100** may take control in certain special circumstances including objects with various properties where DCADO Unit **100** may offer superior performance even though User **50** and/or non-DCADO system may generally be preferred. Once Loader **98***a* leaves such special circumstances, DCADO Unit **100** may release control to User **50** and/or non-DCADO system. In general, DCADO Unit **100** can take control from, share control with, or release control to User **50**, non-DCADO system, and/or other system or process at any time, in any circumstances, and remain in control for any period of time as needed.

In some embodiments, DCADO Unit **100** may control one or more sub-devices, sub-systems, or elements of Loader **98***a* while User **50** and/or non-DCADO system may control other one or more sub-devices, sub-systems, or elements of Loader **98***a*. For example, User **50** and/or non-DCADO system may control the moving system (i.e. drivetrain, powertrain, etc.) of Loader **98***a*, while DCADO Unit **100** may control an arm and bucket of Loader **98***a*. Any other combination of controlling various sub-devices, sub-systems, or elements of Loader **98***a* by DCADO Unit **100** and User **50** and/or non-DCADO system can be implemented.

Referring to FIG. **39**, in some exemplary embodiments, Device **98** may be or include Boat **98***b*. Boat **98***b* may be operated by User **50** in person or remotely. Boat **98***b* may include or be coupled to one or more Sensors **92** (i.e. collectively referred to as Sensor **92**, etc.) such as Camera **92***a*, Microphone **92***b*, Lidar **92***c*, Radar **92***d*, Sonar **92***e*, etc. and/or Object Processing Unit **93** that can detect Objects **615***ba*-**615***bd*, and/or other elements or information in Boat's **98***b* surrounding. Object Processing Unit **93** may include Picture Recognizer **94***a*, Sound Recognizer **94***b*, Lidar Processing Unit **94***c*, Radar Processing Unit **94***d*, Sonar Processing Unit **94***e*, and/or other elements or functionalities as applicable. Object Processing Unit **93** may create or generate one or more (i.e. stream, etc.) Collections of Object Representations **525** comprising Object Representations **625**, Object Properties **630**, and/or other elements or information representing Objects **615** detected in Boat's **98***b* surrounding. Boat **98***b* may also include or be controlled by Logic Circuit **250** (i.e. microcontroller, etc.), Processor **11** (i.e. including any Application Program **18** running thereon, etc.), and/or other processing element that receives User's **50** (i.e. operator's, etc.) operating directions and causes desired operations with Boat **98***b* such as moving, maneuvering, and/or other operations. User **50** can interact with Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element through inputting operating directions via Human-machine Interface **23** such as one or more steering wheels, levers, pedals, buttons, or other input devices. For instance, responsive to User's **50** manipulating a steering wheel and one or more levers, Logic Circuit **250** or Processor **11** may cause one or more motors or other actuators to move or maneuver Boat **98***b*. Boat **98***b* may also include or be coupled to DCADO Unit **100**. DCADO Unit **100** may be embedded (i.e. integrated, etc.) into or coupled to Boat's **98***b* Logic Circuit **250**, Processor **11**, and/or other processing element. DCADO Unit **100** may also be a program embedded (i.e. integrated, etc.) into or interfaced with Application Program **18** running on Processor **11** and/or other processing element. DCADO Unit **100** can obtain Instruction Sets **526** used or executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element. In some aspects, Instruction Sets **526** may include one or more inputs into or outputs from Boat's **98***b* Logic Circuit **250** (i.e. microcontroller, etc.). In other aspects, Instruction Sets **526** may include one or more instruction sets from Boat's **98***b* Processor's **11** registers or

other components. In further aspects, Instruction Sets **526** may include one or more instruction sets used or executed in Application Program **18**. DCADO Unit **100** may also optionally obtain any Extra Info **527** (i.e. time, location, computed, contextual, and/or other information, etc.) related to Boat's **98***b* operation. As User **50** operates Boat **98***b* in circumstances including objects with various properties as shown, DCADO Unit **100** may learn Boat's **98***b* operations in these circumstances by correlating Collections of Object Representations **525** representing Objects **615** detected in Boat's **98***b* surrounding with one or more Instruction Sets **526** used or executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element. Any Extra Info **527** related to Boat's **98***b* operation may also optionally be correlated with Collections of Object Representations **525**. DCADO Unit **100** may store this knowledge into Knowledgebase **530** (i.e. Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.). In the future, DCADO Unit **110** may compare incoming Collections of Object Representations **525** representing Objects **615** detected in Boat's **98***b* surrounding with previously learned Collections of Object Representations **525** including optionally using any Extra Info **527** for enhanced decision making. If substantially similar or at least a partial match is found or determined, the Instruction Sets **526** correlated with the previously learned Collections of Object Representations **525** can be autonomously executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element, thereby enabling autonomous operation of Boat **98***b* in similar circumstances as in previously learned ones. For instance, Boat **98***b* comprising DCADO Unit **100** may learn User **50**-directed moving, maneuvering, and/or other operations in a circumstance that includes Fishing Boat **615***ba*, Lighthouse **615***bb*, Sailboat **615***bc*, Cruise Ship **615***bd*, and/or other Objects **615** among which Boat **98***b* may need to maneuver. In the future, when a circumstance that includes Objects **615** with similar Object Properties **630** is encountered, Boat **98***b* may implement moving, maneuvering, and/or other operations autonomously. In some aspects, the shore (not enumerated) or any part thereof (i.e. cliff, ridge, beach, etc.) may be detected as an Object **615** itself, which may then be learned and used in autonomous operation of Boat **98***b*.

Referring to FIG. **40**, in some exemplary embodiments, an Area of Interest **450** can be utilized. In one example, Area of Interest **450** may include a radial, circular, elliptical, or other such area around Boat **98***b*. In another example, Area of Interest **450** may include a triangular, rectangular, octagonal, or other such area around Boat **98***b*. In a further example, Area of Interest **450** may include a spherical, cubical, pyramid-like, or other such area around Boat **98***b* as applicable to 3D space. Any other Area of Interest **450** shape can be utilized depending on implementation. The shape and/or size of Area of Interest **450** can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. Utilizing Area of Interest **450** enables DCADO Unit **100** to focus on Boat's **98***b* immediate surrounding, thereby avoiding extraneous detail in the rest of the surrounding. In some aspects, Area of Interest **450** can be subdivided into sub-areas (i.e. sub-circles, sub-rectangles, sub-spheres, etc.). Sub-areas can be used to classify the surrounding by distance from Boat **98***b*. For example, the surrounding closer to Boat **98***b* may be more important and may be assigned higher importance index or weight. As User **50** operates Boat **98***b* in circumstances including objects with various properties as shown, DCADO Unit **100** may learn Boat's **98***b* operations in these circumstances by correlating Collections of Object Representations **525** representing Objects **615** detected in Area of Interest **450** around Boat **98***b* with one or more Instruction Sets **526** used or executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element. Any Extra Info **527** related to Boat's **98***b* operation may also optionally be correlated with Collections of Object Representations **525**. DCADO Unit **100** may store this knowledge into Knowledgebase **530** (i.e. Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.). In the future, DCADO Unit **110** may compare incoming Collections of Object Representations **525** representing Objects **615** detected in Area of Interest **450** around Boat **98***b* with previously learned Collections of Object Representations **525** including optionally using any Extra Info **527** for enhanced decision making. If substantially similar or at least a partial match is found or determined, the Instruction Sets **526** correlated with the previously learned Collections of Object Representations **525** can be autonomously executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element, thereby enabling autonomous operation of Boat **98***b* in similar Areas of Interest **450** as in previously learned ones. For instance, Boat **98***b* comprising DCADO Unit **100** may learn User **50**-directed moving, maneuvering, and/or other operations in an Area of Interest **450** that includes Fishing Boat **615***ba*, Lighthouse **615***bb*, Cruise Ship **615***bd*, and/or other Objects **615** among which Boat **98***b* may need to maneuver. In the future, when an Area of Interest **450** that includes Objects **615** with similar Object Properties **630** is encountered, Boat **98***b* may implement moving, maneuvering, and/or other operations autonomously.

The features, functionalities, and embodiments described with respect to Loader **98***a* and Boat **98***b* can be implemented in any situation where Device **98** may need to autonomously maneuver among, interact with, or perform other operations relative to objects in its surrounding. Therefore, the features, functionalities, and embodiments described with respect to Loader **98***a* and Boat **98***b* can similarly be implemented on any computing enabled machine such as a bulldozer, an excavator, a crane, a forklift, a truck, a construction machine, an assembly machine, an object handling machine, an object dispensing machine, a sorting machine, a restocking machine, an industrial machine, an agricultural machine, a harvesting machine, a building control system, a home or other appliance, a toy, a robot, a tank, an aircraft, a vessel, a submarine, a ground vehicle, an aerial vehicle, an aquatic vehicle, and/or other computing-enabled machine or system.

In yet some exemplary embodiments, Device **98** may be or include a control device such as a thermostat, control panel, remote or other controller, and/or other control device. For instance, a thermostat comprising DCADO Unit **100** may learn User's **50** setting temperature of an air conditioning system controlled by the thermostat in a circumstance that includes User **50** and/or other persons entering or being present in a room. In the future, when a circumstance that includes User **50** and/or other persons entering or being present in the room is encountered, thermostat may implement setting temperature of the air conditioning system autonomously. In some aspects, a control device may be included in the device being controlled (i.e. control panel of an oven, refrigerator, fixture, etc.). In other aspects, a control device may be separate from the device

being controlled (i.e. remote controller of a television device, etc.). In yet further exemplary embodiments, Device **98** may be or include a mobile computer such as a smartphone, tablet, and/or other mobile computer. For instance, a smartphone comprising DCADO Unit **100** may learn User **50**-directed playing a music file, setting a vibrate mode, and/or other operations in a circumstance that includes objects with various properties. In the future, when a circumstance that includes objects with similar properties is encountered, smartphone may implement playing music file, setting vibrate mode, and/or other operations autonomously. In general, Device **98** may be or include any movable, stationary, or other device. One of ordinary skill in art will understand that Device **98** may be or include any device that can implement and/or benefit from the functionalities described herein.

It must be noted that as used herein and in the appended claims, the singular forms "a", "an", and "the" include plural referents unless the context clearly dictates otherwise.

A number of embodiments have been described herein. While this disclosure contains many specific implementation details, these should not be construed as limitations on the scope of the disclosure or of what may be claimed, but rather as descriptions of features specific to particular embodiments. It should be understood that various modifications can be made without departing from the spirit and scope of the disclosure. The logic flows depicted in the figures do not require the particular order shown, or sequential order, to achieve desirable results. In addition, other or additional steps, elements, or connections can be included, or some of the steps, elements, or connections can be eliminated, or a combination thereof can be utilized in the described flows, illustrations, or descriptions. Further, the various aspects of the disclosed devices, apparatuses, systems, and/or methods can be combined in whole or in part with each other to produce additional implementations. Moreover, separation of various components in the embodiments described herein should not be understood as requiring such separation in all embodiments, and it should be understood that the described components can generally be integrated together in a single product or packaged into multiple products. Accordingly, other embodiments are within the scope of the following claims.

The invention claimed is:

1. A system comprising:

one or more processors configured to perform at least:

accessing a memory that stores at least a knowledgebase that includes a first circumstance representation correlated with a first one or more instruction sets for operating a first device, wherein the first circumstance representation represents a first circumstance detected at least in part by one or more sensors of the first device, and wherein at least a portion of the first circumstance representation or at least a portion of the first one or more instruction sets for operating the first device is learned in a learning process that includes operating the first device at least partially by a user;

generating or receiving a second circumstance representation, wherein the second circumstance representation represents a second circumstance detected at least in part by: the one or more sensors of the first device, or one or more sensors of a second device;

anticipating the first one or more instruction sets for operating the first device based on at least partial match between the second circumstance representation and the first circumstance representation; and

at least in response to the anticipating, executing the first one or more instruction sets for operating the first device, wherein the first device or the second device autonomously performs one or more operations defined by the first one or more instruction sets for operating the first device.

2. The system of claim **1**, wherein the first one or more instruction sets for operating the first device include one or more information about one or more states of: the first device, or a portion of the first device.

3. The system of claim **1**, wherein the first circumstance representation includes a first one or more object representations, and wherein the second circumstance representation includes a second one or more object representations.

4. The system of claim **3**, wherein the second circumstance representation represents the second circumstance detected at least in part by the one or more sensors of the first device, and wherein the first device autonomously performs the one or more operations defined by the first one or more instruction sets for operating the first device.

5. The system of claim **3**, wherein the second circumstance representation represents the second circumstance detected at least in part by the one or more sensors of the second device, and wherein the second device autonomously performs the one or more operations defined by the first one or more instruction sets for operating the first device.

6. The system of claim **3**, wherein the system further comprising:

a server that receives from the first device at least one of: the first circumstance representation, or the first one or more instruction sets for operating the first device, and wherein the second device receives from the server at least one of: the first circumstance representation, or the first one or more instruction sets for operating the first device, and wherein the second device autonomously performs the one or more operations defined by the first one or more instruction sets for operating the first device.

7. The system of claim **3**, wherein the one or more processors are further configured to perform at least:

modifying: the first one or more instruction sets for operating the first device, or a copy of the first one or more instruction sets for operating the first device, and wherein the anticipating the first one or more instruction sets for operating the first device based on the at least partial match between the second circumstance representation and the first circumstance representation includes: anticipating the modified the first one or more instruction sets for operating the first device based on the at least partial match between the second circumstance representation and the first circumstance representation, or anticipating the modified the copy of the first one or more instruction sets for operating the first device based on the at least partial match between the second circumstance representation and the first circumstance representation, and wherein the executing the first one or more instruction sets for operating the first device includes: executing the modified the first one or more instruction sets for operating the first device, or executing the modified the copy of the first one or more instruction sets for operating the first device, and wherein the autonomously performing, by the first device or by the second device, the one or more operations defined by the first one or more instruction sets for operating the first device includes: autonomously performing, by the first device or by the second device, one or more operations defined by the modified

the first one or more instruction sets for operating the first device, or autonomously performing, by the first device or by the second device, one or more operations defined by the modified the copy of the first one or more instruction sets for operating the first device.

8. The system of claim **3**, wherein the one or more processors are further configured to perform at least:

modifying at least one of: the first circumstance representation, a copy of the first circumstance representation, the second circumstance representation, or a copy of the second circumstance representation, and wherein the at least partial match between the second circumstance representation and the first circumstance representation includes: (i) at least partial match between the modified the second circumstance representation and the first circumstance representation, (ii) at least partial match between the modified the copy of the second circumstance representation and the first circumstance representation, (iii) at least partial match between the second circumstance representation and the modified the first circumstance representation, (iv) at least partial match between the second circumstance representation and the modified the copy of the first circumstance representation, (v) at least partial match between the modified the second circumstance representation and the modified the first circumstance representation, (vi) at least partial match between the modified the copy of the second circumstance representation and the modified the copy of the first circumstance representation, (vii) at least partial match between the modified the second circumstance representation and the modified the copy of the first circumstance representation, or (viii) at least partial match between the modified the copy of the second circumstance representation and the modified the first circumstance representation.

9. The system of claim **3**, wherein the knowledgebase further includes a third circumstance representation correlated with a second one or more instruction sets for operating the first device, and wherein the third circumstance representation represents a third circumstance detected at least in part by the one or more sensors of the first device, and wherein at least a portion of the third circumstance representation or at least a portion of the second one or more instruction sets for operating the first device is learned in another learning process that includes operating the first device at least partially by the user.

10. The system of claim **3**, wherein the knowledgebase further includes a third circumstance representation correlated with a second one or more instruction sets for operating the first device, and wherein the third circumstance representation represents a third circumstance detected at least in part by the one or more sensors of the first device, and wherein at least a portion of the third circumstance representation or at least a portion of the second one or more instruction sets for operating the first device is learned in another learning process that includes operating the first device at least partially by another user.

11. The system of claim **3**, wherein the knowledgebase further includes a third circumstance representation correlated with a second one or more instruction sets for operating a third device, and wherein the third circumstance representation represents a third circumstance detected at least in part by one or more sensors of the third device, and wherein at least a portion of the third circumstance representation or at least a portion of the second one or more instruction sets for operating the third device is learned in another learning process that includes operating the third device at least partially by: the user, or another user.

12. The system of claim **3**, wherein the knowledgebase further includes a third circumstance representation correlated with a second one or more instruction sets for operating a third device, and wherein the third circumstance representation represents a third circumstance detected at least in part by one or more sensors of the third device, and wherein the one or more processors are further configured to perform at least:

generating or receiving a fourth circumstance representation, wherein the fourth circumstance representation represents a fourth circumstance detected at least in part by one or more sensors of a fourth device;

anticipating the second one or more instruction sets for operating the third device based on at least partial match between the fourth circumstance representation and the third circumstance representation; and

at least in response to the anticipating the second one or more instruction sets for operating the third device, executing the second one or more instruction sets for operating the third device, wherein the fourth device autonomously performs one or more operations defined by the second one or more instruction sets for operating the third device.

13. The system of claim **3**, wherein the system further comprising:

an artificial intelligence system, and wherein the learning the at least the portion of the first circumstance representation or the at least the portion of the first one or more instruction sets for operating the first device includes learning, at least in part by the artificial intelligence system, the at least the portion of the first circumstance representation or the at least the portion of the first one or more instruction sets for operating the first device, and wherein the anticipating the first one or more instruction sets for operating the first device includes anticipating, at least in part by the artificial intelligence system, the first one or more instruction sets for operating the first device.

14. The system of claim **13**, wherein the artificial intelligence system includes: one or more inputs for receiving one or more object representations, and one or more outputs for providing one or more instruction sets, and wherein the learning, at least in part by the artificial intelligence system, the at least the portion of the first circumstance representation or the at least the portion of the first one or more instruction sets for operating the first device includes applying the first one or more object representations to the one or more inputs of the artificial intelligence system, and wherein the anticipating, at least in part by the artificial intelligence system, the first one or more instruction sets for operating the first device includes:

applying the second one or more object representations to the one or more inputs of the artificial intelligence system; and

receiving the first one or more instruction sets for operating the first device from the one or more outputs of the artificial intelligence system.

15. The system of claim **13**, wherein the artificial intelligence system includes at least one of: the knowledgebase, or a neural network.

16. The system of claim **3**, wherein the first circumstance representation correlated with the first one or more instruction sets for operating the first device include at least a portion of the first one or more object representations

connected, using at least one or more connections, with at least a portion of the first one or more instruction sets for operating the first device.

17. The system of claim **3**, wherein the anticipating the first one or more instruction sets for operating the first device based on the at least partial match between the second circumstance representation and the first circumstance representation includes determining the first one or more instruction sets for operating the first device based on at least partial match between the second one or more object representations and the first one or more object representations.

18. The system of claim **3**, wherein at least some elements of the system are included in: a single device, or multiple devices, and wherein the one or more processors include: one or more microcontrollers, one or more computing devices, or one or more electronic devices, and wherein the memory includes: a non-transitory machine readable medium, a volatile memory, a non-volatile memory, a storage device, or a storage system, and wherein the one or more sensors of the first device include at least one of: one or more cameras, one or more microphones, one or more radars, one or more lidars, one or more sonars, or one or more apparatuses for detecting objects or object properties, and wherein the one or more sensors of the second device include at least one of: one or more cameras, one or more microphones, one or more radars, one or more lidars, one or more sonars, or one or more apparatuses for detecting objects or object properties, and wherein the first device includes: a robot, a vehicle, an appliance, an electronic device, or a mechanical machine, and wherein the second device includes: a robot, a vehicle, an appliance, an electronic device, or a mechanical machine, and wherein the first circumstance representation is a data structure that includes one or more data about the first circumstance, and wherein the second circumstance representation is a data structure that includes one or more data about the second circumstance, and wherein the first circumstance includes one or more objects detected at least in part by the one or more sensors of the first device at a first time or during a first time period, and wherein the second circumstance includes: one or more objects detected at least in part by the one or more sensors of the first device at a second time or during a second time period, or one or more objects detected at least in part by the one or more sensors of the second device at a second time or during a second time period, and wherein an instruction set of the first one or more instruction sets for operating the first device includes at least one of: only one instruction, multiple instructions, one or more inputs, one or more commands, one or more keywords, one or more symbols, one or more operators, one or more variables, one or more parameters, one or more characters, one or more numbers, one or more values, one or more signals, one or more binary bits, one or more functions, one or more function references, one or more objects, one or more object references, one or more data structures, one or more data structure references, one or more states, one or more representations of one or more states, one or more representations of one or more user inputs, one or more codes, one or more data, or one or more information, and wherein the first one or more object representations include: one object representation, multiple object representations, a collection of object representations, or a stream of collections of object representations, and wherein the second one or more object representations include: one object representation, multiple object representations, a collection of object representations, or a stream of collections of object representations, and wherein the first one or more object representations include: one or more three dimensional representations of one or more objects, one or more digital pictures that depict one or more objects, one or more digital pictures that depict one or more representations of one or more objects, one or more information about one or more properties of one or more objects, or one or more computer representations of one or more objects, and wherein the second one or more object representations include: one or more three dimensional representations of one or more objects, one or more digital pictures that depict one or more objects, one or more digital pictures that depict one or more representations of one or more objects, one or more information about one or more properties of one or more objects, or one or more computer representations of one or more objects, and wherein the at least the portion of the first circumstance representation includes: at least a portion of one object representation of the first one or more object representations, multiple portions of multiple object representations of the first one or more object representations, or the entire first circumstance representation, and wherein the at least the portion of the first one or more instruction sets for operating the first device includes: at least a portion of one instruction set of the first one or more instruction sets for operating the first device, multiple portions of multiple instruction sets of the first one or more instruction sets for operating the first device, or the entire first one or more instruction sets for operating the first device, and wherein: an object of the first circumstance is the same as an object of the second circumstance, multiple objects of the first circumstance are the same as multiple objects of the second circumstance, all objects of the first circumstance are the same as all objects of the second circumstance, or all objects of the first circumstance are different than all objects of the second circumstance.

19. A method implemented using a computing system that includes one or more processors, the method comprising:

accessing a memory that stores at least a knowledgebase that includes a first circumstance representation correlated with a first one or more instruction sets for operating a first device, wherein the first circumstance representation represents a first circumstance detected at least in part by one or more sensors of the first device, and wherein at least a portion of the first circumstance representation or at least a portion of the first one or more instruction sets for operating the first device is learned in a learning process that includes operating the first device at least partially by a user;

generating or receiving a second circumstance representation, wherein the second circumstance representation represents a second circumstance detected at least in part by: the one or more sensors of the first device, or one or more sensors of a second device;

anticipating the first one or more instruction sets for operating the first device based on at least partial match between the second circumstance representation and the first circumstance representation; and

at least in response to the anticipating, executing the first one or more instruction sets for operating the first device, wherein the first device or the second device autonomously performs one or more operations defined by the first one or more instruction sets for operating the first device.

20. A system comprising:

means for processing; and

a memory that stores at least a knowledgebase that includes a first circumstance representation correlated with a first one or more instruction sets for operating a first device, wherein the first circumstance representa-

tion represents a first circumstance detected at least in part by one or more sensors of the first device, and wherein at least a portion of the first circumstance representation or at least a portion of the first one or more instruction sets for operating the first device is learned in a learning process that includes operating the first device at least partially by a user, and wherein the means for processing is configured to perform at least:

generating or receiving a second circumstance representation, wherein the second circumstance representation represents a second circumstance detected at least in part by: the one or more sensors of the first device, or one or more sensors of a second device;

anticipating the first one or more instruction sets for operating the first device based on at least partial match between the second circumstance representation and the first circumstance representation; and

at least in response to the anticipating, executing the first one or more instruction sets for operating the first device, wherein the first device or the second device autonomously performs one or more operations defined by the first one or more instruction sets for operating the first device.

* * * * *

Exhibit C

US010607134B1

(12) **United States Patent**
Cosic

(10) **Patent No.: US 10,607,134 B1**
(45) **Date of Patent: *Mar. 31, 2020**

(54) **ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING AN AVATAR'S CIRCUMSTANCES FOR AUTONOMOUS AVATAR OPERATION**

(71) Applicant: **Jasmin Cosic**, Miami, FL (US)

(72) Inventor: **Jasmin Cosic**, Miami, FL (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 763 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: **15/382,743**

(22) Filed: **Dec. 19, 2016**

(51) **Int. Cl.**
G06N 3/00 (2006.01)
G06N 20/00 (2019.01)

(52) **U.S. Cl.**
CPC ***G06N 3/006*** (2013.01); ***G06N 20/00*** (2019.01)

(58) **Field of Classification Search**
CPC G06N 3/006
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,370,707 A 1/1983 Phillips et al.
4,730,315 A 3/1988 Saito et al.
4,860,203 A 8/1989 Corrigan et al.
5,560,011 A 9/1996 Uyama
5,592,661 A 1/1997 Eisenberg et al.
5,602,982 A 2/1997 Judd et al.
5,758,333 A 5/1998 Bauer et al.
5,983,232 A 11/1999 Zhang
6,026,234 A 2/2000 Hanson et al.
6,088,731 A 7/2000 Kiraly et al.
6,106,299 A 8/2000 Ackermann et al.
6,126,330 A 10/2000 Knight
6,314,558 B1 11/2001 Angel et al.
6,643,842 B2 11/2003 Angel et al.
6,728,689 B1 4/2004 Drissi et al.
6,735,632 B1 5/2004 Kiraly et al.
6,754,631 B1 6/2004 Din
6,801,912 B2 10/2004 Moskowitz et al.
6,842,877 B2 1/2005 Robarts et al.
6,850,942 B2 2/2005 Cotner et al.
6,915,105 B2 7/2005 Masuda
6,973,446 B2 12/2005 Mamitsuka et al.
7,017,153 B2 3/2006 Gouriou et al.
7,052,277 B2 5/2006 Kellman
7,082,435 B1 7/2006 Guzman et al.
7,113,946 B2 9/2006 Cosic
7,117,225 B2 10/2006 Cosic
(Continued)

OTHER PUBLICATIONS

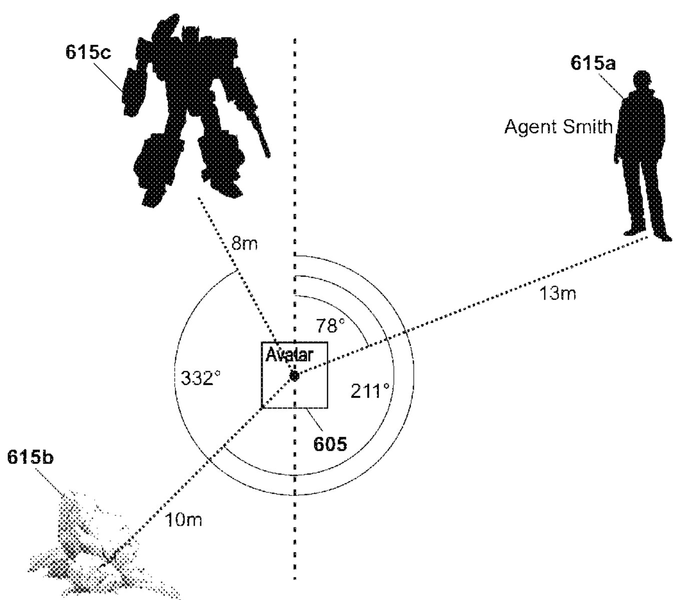
Zhang, Yiying, Lei Guo, and Nicolas D. Georganas. "AGILE: An architecture for agent-based collaborative and interactive virtual environments." Proc. Workshop on Application Virtual Reality Technologies for Future Telecommunication System, IEEE Globecom '2000 Conference. 2000. (Year: 2000).*
(Continued)

Primary Examiner — Hal Schnee

(57) **ABSTRACT**

Aspects of the disclosure generally relate to computing devices and/or systems, and may be generally directed to devices, systems, methods, and/or applications for learning an avatar's or an application's operation in various circumstances, storing this knowledge in a knowledgebase (i.e. neural network, graph, sequences, etc.), and/or enabling autonomous operation of the avatar or the application.

20 Claims, 39 Drawing Sheets



(56) **References Cited**

U.S. PATENT DOCUMENTS

7,222,127 B1 5/2007 Bem et al.
7,240,335 B2 7/2007 Angel et al.
7,249,349 B2 7/2007 Hundt et al.
7,395,255 B2 7/2008 Li
7,409,401 B2 8/2008 Hansen et al.
7,424,705 B2 9/2008 Lewis et al.
7,444,338 B1 10/2008 Fisher
7,478,371 B1 1/2009 Gove
7,484,205 B2 1/2009 Venkatapathy
7,533,128 B1 5/2009 Sanchez et al.
7,565,340 B2 7/2009 Herlocker et al.
7,721,218 B2 5/2010 Awe et al.
7,765,537 B2 7/2010 Havin et al.
7,797,259 B2 9/2010 Jiang et al.
7,831,564 B1 11/2010 Wei et al.
7,840,060 B2 11/2010 Podilchuk
7,849,114 B2 12/2010 Boss et al.
7,925,984 B2 4/2011 Awe et al.
7,987,144 B1 7/2011 Drissi et al.
8,005,828 B2 8/2011 Buchner et al.
8,019,699 B2 9/2011 Baxter
8,078,556 B2 12/2011 Adi et al.
8,090,669 B2 1/2012 Shahani et al.
8,137,112 B2 3/2012 Woolf et al.
8,166,463 B2 4/2012 Gill et al.
8,195,674 B1 6/2012 Bem et al.
8,196,119 B2 6/2012 Gill et al.
8,244,730 B2 8/2012 Gupta
8,251,704 B2 8/2012 Woolf et al.
8,261,199 B2 9/2012 Cradick et al.
8,266,608 B2 9/2012 Hecht et al.
8,335,805 B2 12/2012 Cosic
8,364,612 B2 1/2013 Van Gael et al.
8,386,401 B2 2/2013 Virkar et al.
8,397,227 B2 3/2013 Fan et al.
8,417,740 B2 4/2013 Cosic
8,464,225 B2 6/2013 Greifeneder
8,549,359 B2 10/2013 Zheng
8,572,035 B2 10/2013 Cosic
8,589,414 B2 11/2013 Waite et al.
8,655,260 B2 2/2014 Yueh et al.
8,655,900 B2 2/2014 Cosic
8,667,472 B1 3/2014 Molinari
8,762,428 B2 6/2014 Kulack et al.
8,949,186 B1 2/2015 Yueh et al.
8,996,432 B1 3/2015 Fu
9,047,324 B2 6/2015 Cosic
9,268,454 B2 2/2016 Hamilton, II et al.
9,282,309 B1 3/2016 Cosic
9,298,749 B2 3/2016 Cosic
9,305,216 B1 4/2016 Mishra
9,367,806 B1 6/2016 Cosic
9,443,192 B1 9/2016 Cosic
2002/0019822 A1 2/2002 Seki et al.
2003/0026588 A1 2/2003 Elder et al.
2003/0065662 A1 4/2003 Cosic
2004/0117771 A1 6/2004 Venkatapathy
2004/0194017 A1 9/2004 Cosic
2004/0249774 A1 12/2004 Caid et al.
2004/0267521 A1 12/2004 Cutler et al.
2005/0149517 A1 7/2005 Cosic
2005/0149542 A1 7/2005 Cosic
2005/0154745 A1 7/2005 Hansen et al.
2005/0240412 A1 10/2005 Fujita
2005/0245303 A1 11/2005 Graepel et al.
2005/0289105 A1 12/2005 Cosic
2006/0047612 A1 3/2006 Stanley et al.
2006/0184410 A1 8/2006 Ramamurthy et al.
2006/0190930 A1 8/2006 Hecht et al.
2006/0259466 A1 11/2006 Bilotti
2006/0265406 A1 11/2006 Chkodrov et al.
2007/0006159 A1 1/2007 Hecht et al.
2007/0050606 A1 3/2007 Ferren et al.
2007/0050719 A1 3/2007 Lui et al.
2007/0058856 A1 3/2007 Boregowda et al.
2007/0061735 A1 3/2007 Hoffberg et al.
2007/0106633 A1 5/2007 Reiner
2008/0071770 A1 3/2008 Schloter et al.
2008/0144893 A1 6/2008 Guo et al.
2008/0215508 A1 9/2008 Hanneman et al.
2008/0254429 A1 10/2008 Woolf et al.
2008/0281764 A1 11/2008 Baxter
2008/0288259 A1 11/2008 Chambers et al.
2009/0044113 A1 2/2009 Jones et al.
2009/0067727 A1 3/2009 Hirohata
2009/0070337 A1 3/2009 Romem et al.
2009/0110061 A1 4/2009 Lee et al.
2009/0131152 A1 5/2009 Busse
2009/0136095 A1 5/2009 Marcon et al.
2009/0141969 A1 6/2009 Yu et al.
2009/0222388 A1 9/2009 Hua et al.
2009/0287643 A1 11/2009 Corville et al.
2009/0324010 A1 12/2009 Hou
2010/0023541 A1 1/2010 Cosic
2010/0033780 A1 2/2010 Gitter
2010/0063949 A1 3/2010 Eaton et al.
2010/0082536 A1 4/2010 Cosic
2010/0114746 A1 5/2010 Bobbitt et al.
2010/0138370 A1 6/2010 Wu et al.
2010/0241595 A1 9/2010 Felsher
2010/0278420 A1 11/2010 Shet et al.
2011/0004586 A1 1/2011 Cherryholmes et al.
2011/0007079 A1 1/2011 Perez et al.
2011/0030031 A1 2/2011 Lussier et al.
2011/0085734 A1 4/2011 Berg et al.
2011/0093435 A1 4/2011 Zha et al.
2011/0218672 A1 9/2011 Maisonnier et al.
2011/0270794 A1 11/2011 Drory et al.
2012/0089570 A1 4/2012 Zha et al.
2012/0150773 A1 6/2012 DiCorpo et al.
2012/0167057 A1 6/2012 Schmich et al.
2012/0284026 A1 11/2012 Cardillo et al.
2012/0290347 A1 11/2012 Elazouni et al.
2013/0007532 A1 1/2013 Miller et al.
2013/0156345 A1 6/2013 Shmunk
2013/0159021 A1 6/2013 Felsher
2013/0204907 A1 8/2013 Alonso Alarcon et al.
2013/0218932 A1 8/2013 Cosic
2013/0226974 A1 8/2013 Cosic
2013/0238533 A1 9/2013 Virkar et al.
2013/0278631 A1 10/2013 Border et al.
2014/0052717 A1 2/2014 Franks et al.
2014/0075249 A1 3/2014 Sato et al.
2014/0143276 A1 5/2014 Rogers et al.
2014/0161250 A1 6/2014 Leeds et al.
2014/0164430 A1 6/2014 Hadjieleftheriou et al.
2014/0177946 A1 6/2014 Lim et al.
2014/0207580 A1 7/2014 Minnis et al.
2014/0211988 A1 7/2014 Fan et al.
2015/0006171 A1 1/2015 Westby et al.
2015/0039304 A1 2/2015 Wein
2015/0055821 A1 2/2015 Fotland
2015/0264306 A1 9/2015 Marilly et al.
2015/0269415 A1 9/2015 Gelbman
2015/0310041 A1 10/2015 Kier et al.
2015/0324685 A1 11/2015 Bohn et al.
2015/0339213 A1 11/2015 Lee et al.
2016/0140999 A1 5/2016 Cosic
2016/0142650 A1 5/2016 Cosic
2016/0167226 A1 6/2016 Schnittman
2016/0246819 A1 8/2016 Cosic
2016/0246850 A1 8/2016 Cosic
2016/0246868 A1 8/2016 Cosic
2016/0274187 A1 9/2016 Menon et al.
2016/0292185 A1 10/2016 Cosic
2016/0328480 A1 11/2016 Owens et al.

OTHER PUBLICATIONS

Eladhari, Mirjam P., "Semi-Autonomous Avatars in Virtual Game Worlds," Pre-conference to the ECREA 2010—3rd European Communication Conference, Avatars and Humans. Representing Users

(56) **References Cited**

OTHER PUBLICATIONS

in Digital Games, Hamburg, Germany, Oct. 2010, Available from: Jan. 21, 2012 (Year: 2012).*
About | OpenCV, retrieved from <URL: http://opencv.org/about.html> on Dec. 13, 2014, 1 pages.
Animetrics, Inc. | 3D Facial Recognition, retrieved from <URL: http://animetrics.com/> on Dec. 13, 2014, 2 pages.
Bag-of-words model, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 2 pages.
Beier-Neely morphing algorithm, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 1 pages.
Computer vision, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 7 pages.
Convolutional neural network, retrieved from <URL: http://wikipedia.com> on Nov. 11, 2015, 5 pages.
Decimation (signal processing), retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 3 pages.
Digital image processing, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 3 pages.
Dissolve (filmmaking), retrieved from <URL: http://wikipedia.com> on Nov. 11, 2015, 2 pages.
Facial recognition system, retrieved from <URL: http://wikipedia.com> on Nov. 11, 2015, 6 pages.
Feature detection (computer vision), retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 3 pages.
Feature extraction, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 2 pages.
Gesture recognition, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 5 pages.
GrabCut, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 1 pages.
Image processing, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 2 pages.
Image segmentation, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 12 pages.
Image warping, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 2 pages.
Inbetweening, retrieved from <URL: http://wikipedia.com> on Nov. 11, 2015, 2 pages.
Interpolation, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 5 pages.
Language model, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 4 pages.
List of speech recognition software, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 4 pages.
Livewire Segmentation Technique, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 2 pages.
Morphing, retrieved from <URL: http://wikipedia.com> on Nov. 11, 2015, 2 pages.
Motion estimation, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 2 pages.
Muse: Face Morph || Mesh Warping, retrieved from <URL: http://alexwolfe.blogspot.com/2011/10/face-morph-mesh-warping.html> on Nov. 19, 2015, 2 pages.
n-gram, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 5 pages.
Optical flow, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 4 pages.
Outline of object recognition, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 7 pages.
Phonotactics, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 3 pages.
Recurrent neural network, retrieved from <URL: http://wikipedia.com> on Nov. 11, 2015, 5 pages.
Sample rate conversion, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 2 pages.
Simple interactive object extraction, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 2 pages.
Speech recognition, retrieved from <URL: http://wikipedia.com> on Nov. 11, 2015, 12 pages.
Speech segmentation, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 4 pages.
Welcome to recognize-speech.com, retrieved from <URL: http://recognize-speech.com/> on Oct. 18, 2015, 1 pages.
Introduction Speech, retrieved from <URL: http://recognize-speech.com/speech> on Oct. 18, 2015, 1 pages.
Preprocessing, retrieved from <URL: http://recognize-speech.com/preprocessing> on Oct. 18, 2015, 4 pages.
Feature Extraction, retrieved from <URL: http://recognize-speech.com/feature-extraction> on Oct. 18, 2015, 3 pages.
Acoustic model, retrieved from <URL: http://recognize-speech.com/acoustic-model> on Oct. 18, 2015, 2 pages.
Video content analysis, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 2 pages.
Video tracking, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 2 pages.
Andrej Karpathy, Thomas Leung, George Toderici, Rahul Sukthankar, Sanketh Shetty, Li Fei-Fei, Large-scale Video Classification with Convolutional Neural Networks, Apr. 14, 2014, 8 pages, Stanford University.
Karen Simonyan, Andrew Zisserman, Two-Stream Convolutional Networks for Action Recognition in Videos, Nov. 13, 2014, 11 pages, University of Oxford.
About Event Tracing, retrieved from <URL: http://msdn.microsoft.com/en-us/library/aa363668(d=default,I=en-us, v=vs.85).aspx> on Jan. 12, 2014, 2 pages.
Add dynamic Java code to your application, retrieved from <URL: http://www.javaworld.com/article/2071777/design-patterns/add-dynamic-java-code-to-your-application.html> on May 13, 2014, 6 pages.
BCEL, retrieved from <URL: http://commons.apache.org> on May 13, 2014, 2 pages.
Artificial intelligence, retrieved from <URL: http://wikipedia.com> on May 21, 2014, 28 pages.
Artificial neural network, retrieved from <URL: http://wikipedia.com> on May 21, 2014, 12 pages.
Branch (computer science), retrieved from <URL: http://wikipedia.com> on May 6, 2014, 2 pages.
Branch trace, retrieved from <URL: http://wikipedia.com> on May 28, 2014, 2 pages.
C Function Call Conventions and the Stack, retrieved from <URL: http://www.csee.umbc.edu/~chang/cs313.s02/stack.shtml> on Jan. 11, 2014, 4 pages.
Call stack, retrieved from <URL: http://wikipedia.com> on Jan. 11, 2014, 5 pages.
Program compilation and execution flow, retrieved from <URL: http://cs.stackexchange.com/questions/6187/program-compilation-and-execution-flow> on Jan. 9, 2014, 2 pages.
Control unit, retrieved from <URL: http://wikipedia.com> on May 25, 2014, 2 pages.
Create dynamic applications with javax.tools, retrieved from <URL: http://www.ibm.com/developerworks/library/j-jcomp/> on May 13, 2014, 11 pages.
Creating a Debugging and Profiling Agent with JVMTI, retrieved from <URL: http://www.oracle.com/technetwork/articles/java/jvmti-136367.html> on Jan. 11, 2014, 18 pages.
Decision tree learning, retrieved from <URL: http://wikipedia.com> on Jun. 24, 2014, 5 pages.
Dynamic Bytecode Instrumentation, retrieved from <URL: http://www.drdobbs.com/tools/dynamic-bytecode-instrumentation/184406433> on Apr. 26, 2014, 6 pages.
Dynamic programming language, retrieved from <URL: http://wikipedia.com> on May 7, 2014, 4 pages.
Dynamic recompilation, retrieved from <URL: http://wikipedia.com> on May 19, 2014, 4 pages.
eval(), retrieved from <URL: https://developer.mozilla.org/en-US/docsWeb/JavaScript/Reference/Global_Objects/eval> on May 7, 2014, 13 pages.
Function, retrieved from <URL: https://developer.mozilla.org/en-US/docsWeb/JavaScript/Reference/Global_Objects/Function> on May 7, 2014, 11 pages.

(56) **References Cited**

OTHER PUBLICATIONS

FunctionEnter2 Function, retrieved from <URL: https://msdn.microsoft.com/en-us/library/aa964981(d=default,I=en-us,v=vs.110).aspx> on Mar. 19, 2014, 2 pages.

FunctionLeave2 Function, retrieved from <URL: https://msdn.microsoft.com/en-us/library/aa964942(d=default,I=en-us,v=vs.110).aspx> on Mar. 19, 2014, 1 pages.

Heuristic, retrieved from <URL: http://wikipedia.com> on May 22, 2014, 5 pages.

How to: Configure Trace Switches, retrieved from <URL: https://msdn.microsoft.com/en-us/library/vstudio/t06xyy08(d=default,I=en-us,v=vs.100).aspx> on Mar. 19, 2014, 3 pages.

Indirect branch, retrieved from <URL: http://wikipedia.com> on May 5, 2014, 1 pages.

Instruction cycle, retrieved from <URL: http://wikipedia.com> on Apr. 30, 2014, 2 pages.

Instruction set, retrieved from <URL: http://wikipedia.com> on Apr. 29, 2014, 6 pages.

Interrupt, retrieved from <URL: http://wikipedia.com> on May 15, 2014, 5 pages.

Introduction to Instrumentation and Tracing, retrieved from <URL: http://msdn.microsoft.com/en-us/library/aa983649(d=default,I=en-us,v=vs.71).aspx> on Jan. 10, 2014, 2 pages.

Java bytecode, retrieved from <URL: http://wikipedia.com> on May 27, 2014, 4 pages.

Package java.util.logging, retrieved from <URL: http://docs.oracle.com/javase/1.5.0/docs/api/java/util/logging/package-summary.html> on Mar. 19, 2014, 2 pages.

iavac—Java programming language compiler, retrieved from <URL: http://docs.oracle.com/javase/7/docs/technotes/tools/windows/javac.html> on May 13, 2014, 12 pages.

Interface JavaCompiler, retrieved from <URL: http://wikipedia.com> on May 13, 2014, 4 pages.

Javassist, retrieved from <URL: http://wikipedia.com> on May 13, 2014, 2 pages.

Just-in-time compilation, retrieved from <URL: http://wikipedia.com> on May 19, 2014, 5 pages.

JVM Tool Interface, retrieved from <URL: http://docs.oracle.com/javase/7/docs/platform/jvmti/jvmti.html> on Apr. 26, 2014, 127 pages.

Machine code, retrieved from <URL: http://wikipedia.com> on Apr. 30, 2014, 3 pages.

Metaprogramming, retrieved from <URL: http://wikipedia.com> on May 8, 2014, 3 pages.

Microsoft.VisualBasic.Logging Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/ms128122(d=default,I=en-us,v=vs.110).aspx> on Jan. 13, 2014, 1 pages.

Pin (computer program), retrieved from <URL: http://wikipedia.com> on May 6, 2014, 3 pages.

Processor register, retrieved from <URL: http://wikipedia.com> on Apr. 29, 2014, 4 pages.

Profiling (computer programming), retrieved from <URL: http://wikipedia.com> on Jan. 10, 2014, 4 pages.

Profiling Overview, retrieved from <URL: http://msdn.microsoft.com/en-us/library/bb384493(d=default,I=en-us,v=vs.110).aspx> on Jan. 12, 2014, 6 pages.

Program counter, retrieved from <URL: http://wikipedia.com> on Apr. 24, 2014, 2 pages.

ptrace, retrieved from <URL: http://wikipedia.com> on Apr. 24, 2014, 2 pages.

Reflection (computer programming), retrieved from <URL: http://wikipedia.com> on May 7, 2014, 5 pages.

Class Runtime, retrieved from <URL: http://docs.oracle.com/javase/7/docs/api/java/lang/Runtime.html> on Jan. 10, 2014,10 pages.

Self-modifying code, retrieved from <URL: http://wikipedia.com> on Jan. 12, 2014, 6 pages.

Subjective logic, retrieved from <URL: http://wikipedia.com> on May 22, 2014, 5 pages.

System.CodeDom.Compiler Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/z6b99ydtd=default,I=en-us,v=vs.110).aspx> on May 13, 2014, 2 pages.

System.Reflection.Emit Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/xd5fw18y(d=default,=en-us,v=vs.110).aspx> on May 13, 2014, 2 pages.

Chen et al. Case-Based Reasoning System and Artificial Neural Networks: A Review Neural Comput & Applic (2001) 10: pp. 264-276, 13 pages.

John J. Grefenstette, Connie Loggia Ramsey, Alan C. Schultz, Learning Sequential Decision Rules Using Simulation Models and Competition, 1990, Navy Center for Applied Research in Artificial Intelligence, Naval Research Laboratory, Washington, DC, 27 pages.

Alan C. Schultz, John J. Grefenstette, Using a Genetic Algorithm to Learn Behaviors for Autonomous Vehicles, 1992, Navy Center for Applied Research in Artificial Intelligence, Naval Research Laboratory, Washington, DC, 12 pages.

Koppula et al., "Anticipating human activities using object affordances for reactive robotic response", IEEE TRAMI 2016, published May 5, 2015, 16 pages.

Orme, "System design tips for entry level smartphones—part 3", found online at "https://community.arm.com/processors/b/blog/posts/system-design-tips-for-entry-level-smartphones---part-3", Oct. 21, 2013, 12 pages.

Nowostawski, Mariusz et al., "Dynamic Demes Parallel Genetic Algorithm", May 13, 1999, 6 pages.

Stack Overflow, How can I quantify difference between two images?, accessed Aug. 2, 2017 at https://stackoverflow.com/questions/189943/how-can-i-quantify-difference-between-two-images, 8 pages.

Skolicki, Zbigniew et al., "The Influence of Migration Sizes and Intervals on Island Models", Jun. 29, 2005, 8 pages.

Vahid Lad, et al., "Decentralized dynamic resource management support for massively parallel processor arrays", Sep. 11, 2011, 8 pages.

Vahid Lari, et al., "Distributed resource reservation in massively parallel processor arrays", May 16, 2011, 4 pages.

Mohsen Flayati and Yazdan Shirvany, "Artificial Neural Network Approach for Short Term Load Forecasting for Illam Region", Jan. 2007, 5 pages.

Hasim Sak, Andrew Senior, and Francoise Beaufays, "Long Short-Term Memory Recurrent Neural Network Architectures for Large Scale Acoustic Modeling", Jan. 2014, 5 pages.

Jorg Walter and Klaus Schulten, "Implementation of self-organizing neural networks for visuo-motor control of an industrial robot", Jan. 1993, 10 pages.

Heikki Hyotyniemi and Aamo Lehtola, "A Universal Relation Database Interface for Knowledge Based Systems", Apr. 1991, 5 pages.

Mrissa, Michael, et al. "An avatar architecture for the web of things." IEEE Internet Computing 19.2 (2015): 30-38., 9 pages.

Luck, Michael, and Ruth Aylett. "Applying artificial intelligence to virtual reality: Intelligent virtual environments." Applied Artificial Intelligence 14.1 (2000): 3-32., 30 pages.

Terdjimi, Mehdi, et al. "An avatar-based adaptation workflow for the web of things." Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), 2016 IEEE 25th International Conference on. IEEE, 2016., 6 pages.

Bogdanovych, Anton, et al. "Authentic interactive reenactment of cultural heritage with 3D virtual worlds and artificial intelligence." Applied Artificial Intelligence 24.6 (2010): 617-647., 32 pages.

Hernandez, Marco E. Perez, and Stephan Reiff-Marganiec. "Autonomous and self controlling smart objects for the future internet." Future Internet of things and cloud (FiCloud), 2015 3rd international conference on. IEEE, 2015., 8 pages.

Medini et al., "Building a Web of Things with Avatars", Managing the Web of Things (2017), 30 pages.

Tracing (software), retrieved from <URL: http://wikipedia.com> on Jan. 10, 2014, 3 pages.

Tree (data structure), retrieved from <URL: http://wikipedia.com> on Jun. 24, 2014, 6 pages.

PTRACE(2), retrieved from <URL: http://unixhelp.ed.ac.uk/CGI/man-cgi?ptrace> on Mar. 19, 2014, 5 pages.

(56) **References Cited**

OTHER PUBLICATIONS

Wevtutil, retrieved from <URL: http://technet.microsoft.com/en-us/library/cc732848(d=default,I=en-us,v=ws.11).aspx> on Apr. 28, 2014, 5 pages.
Intel Processor Trace, retrieved from <URL: https://software.intel.com/en-us/blogs/2013/09/18/processor-tracing> on Apr. 28, 2014, 3 pages.
Younghoon Jung, Java Dynamics Reflection and a lot more, Oct. 10, 2012, 55 pages, Columbia University.
Amitabh Srivastava, Alan Eustace, Atom A System for Building Customized Program Analysis Tools, May 3, 2004,12 pages.
Mathew Smithson, Kapil Anand, Aparna Kotha, Khaled Elwazeer, Nathan Giles, Rajeev Barua, Binary Rewriting without Relocation Information, Nov. 10, 2010, 11 pages, University of Maryland.
Marek Olszewski, Keir Mierte, Adam Czajkowski, Angela Demle Brown, JIT Instrumentation—A Novel Approach to Dynamically Instrument Operating Systems, Feb. 12, 2007, 14 pages, University of Toronto.
2D computer graphics, retrieved from <URL: http://wikipedia.com> on Nov. 9, 2014, 7 pages.
3D computer graphics, retrieved from <URL: http://wikipedia.com> on Nov. 9, 2014, 3 pages.
3D modeling, retrieved from <URL: http://wikipedia.com> on Nov. 9, 2014, 5 pages.
8: Mouse Handling, retrieved from <URL: http://ericsink.com/wpf3d/8_Mouse.html> on Nov. 13, 2014, 2 pages.
AutoCAD, retrieved from <URL: http://wikipedia.com> on Nov. 9, 2014, 7 pages.
Collision detection, retrieved from <URL: http://wikipedia.com> on Nov. 23, 2014, 5 pages.
Comparison of OpenGL and Direct3D, retrieved from <URL: http://wikipedia.com> on Nov. 22, 2014, 7 pages.
Data dictionary, retrieved from <URL: http://wikipedia.com> on Nov. 6, 2014, 3 pages.
DBA_TABLES, retrieved from <URL: http://wikipedia.com> on Nov. 6, 2014, 1 pages.
Digital sculpting, retrieved from <URL: http://wikipedia.com> on Nov. 12, 2014, 3 pages.
Facial recognition system, retrieved from <URL: http://wikipedia.com> on Dec. 13, 2014, 7 pages.
Game engine, retrieved from <URL: http://wikipedia.com> on Nov. 23, 2014, 4 pages.
JavaFX, retrieved from <URL: http://wikipedia.com> on Nov. 22, 2014, 7 pages.
Jreality, retrieved from <URL: http://wikipedia.com> on Nov. 22, 2014, 3 pages.
List of 14+ Image Recognition APIs, retrieved from <URL: http://blog.mashape.com/list-of-14-image-recognition-apis/> on Dec. 13, 2014, 2 pages.
List of 50+ Face Detection / Recognition APIs, libraries, and software, retrieved from <URL: http://blog.mashape.com/list-of-50-face-detection-recognition-apis/> on Dec. 13, 2014, 4 pages.
List of file formats, retrieved from <URL: http://wikipedia.com> on Nov. 9, 2014, 33 pages.
List of game engines, retrieved from <URL: http://wikipedia.com> on Nov. 22, 2014, 17 pages.
Lua (programming language), retrieved from <URL: http://wikipedia.com> on Nov. 25, 2014, 10 pages.
Mantle (API), retrieved from <URL: http://wikipedia.com> on Nov. 22, 2014, 5 pages.
Microsoft XNA, retrieved from <URL: http://wikipedia.com> on Nov. 22, 2014, 7 pages.
3.4 Getting Information About Databases and Tables, retrieved from <URL: http://dev.mysql.com/doc/refman/5.7/en/getting-information.html> on Nov. 6, 2014, 2 pages.
MySQL show users—how to show the users in a MySQL database, retrieved from <URL: http://alvinalexander.com/blog/post/mysql/show-users-i-ve-created-in-mysql-database> on Nov. 6, 2014, 4 pages.
OpenCV, retrieved from <URL: http://wikipedia.com> on Dec. 13, 2014, 3 pages.
Optical character recognition, retrieved from <URL: http://wikipedia.com> on Dec. 13, 2014, 6 pages.
Outline of object recognition, retrieved from <URL: http://wikipedia.com> on Dec. 13, 2014, 8 pages.
Particle system, retrieved from <URL: http://wikipedia.com> on Nov. 12, 2014, 3 pages.
Physics engine, retrieved from <URL: http://wikipedia.com> on Nov. 23, 2014, 5 pages.
Polygonal modeling, retrieved from <URL: http://wikipedia.com> on Nov. 12, 2014, 4 pages.
POV-Ray, retrieved from <URL: http://wikipedia.com> on Nov. 10, 2014, 5 pages.
Raster graphics, retrieved from <URL: http://wikipedia.com> on Nov. 9, 2014, 3 pages.
Scene graph, retrieved from <URL: http://wikipedia.com> on Nov. 22, 2014, 4 pages.
STL (file format), retrieved from <URL: http://wikipedia.com> on Nov. 11, 2014, 4 pages.
Subdivision surface, retrieved from <URL: http://wikipedia.com> on Nov. 12, 2014, 3 pages.
sysobjects, retrieved from <URL: http://technet.microsoft.com/en-us/library/aa260447(d=default,I=en-us,v=sql.80).aspx> on Nov. 6, 2014, 2 pages.
Vector graphics, retrieved from <URL: http://wikipedia.com> on Nov. 9, 2014, 5 pages.
Vertex (computer graphics), retrieved from <URL: http://wikipedia.com> on Nov. 9, 2014, 2 pages.
Voxel, retrieved from <URL: http://wikipedia.com> on Nov. 11, 2014, 5 pages.
Wavefront .obj file, retrieved from <URL: http://wikipedia.com> on Nov. 9, 2014, 6 pages.
Acrobat 3D tutorials—basic interaction, retrieved from <URL: https://acrobatusers.com/assets/collections/tutorials/legacy/tech_corners/3d/3d_tutorials/basic_interaction.pdf> on Nov. 23, 2014, 2 pages.

* cited by examiner

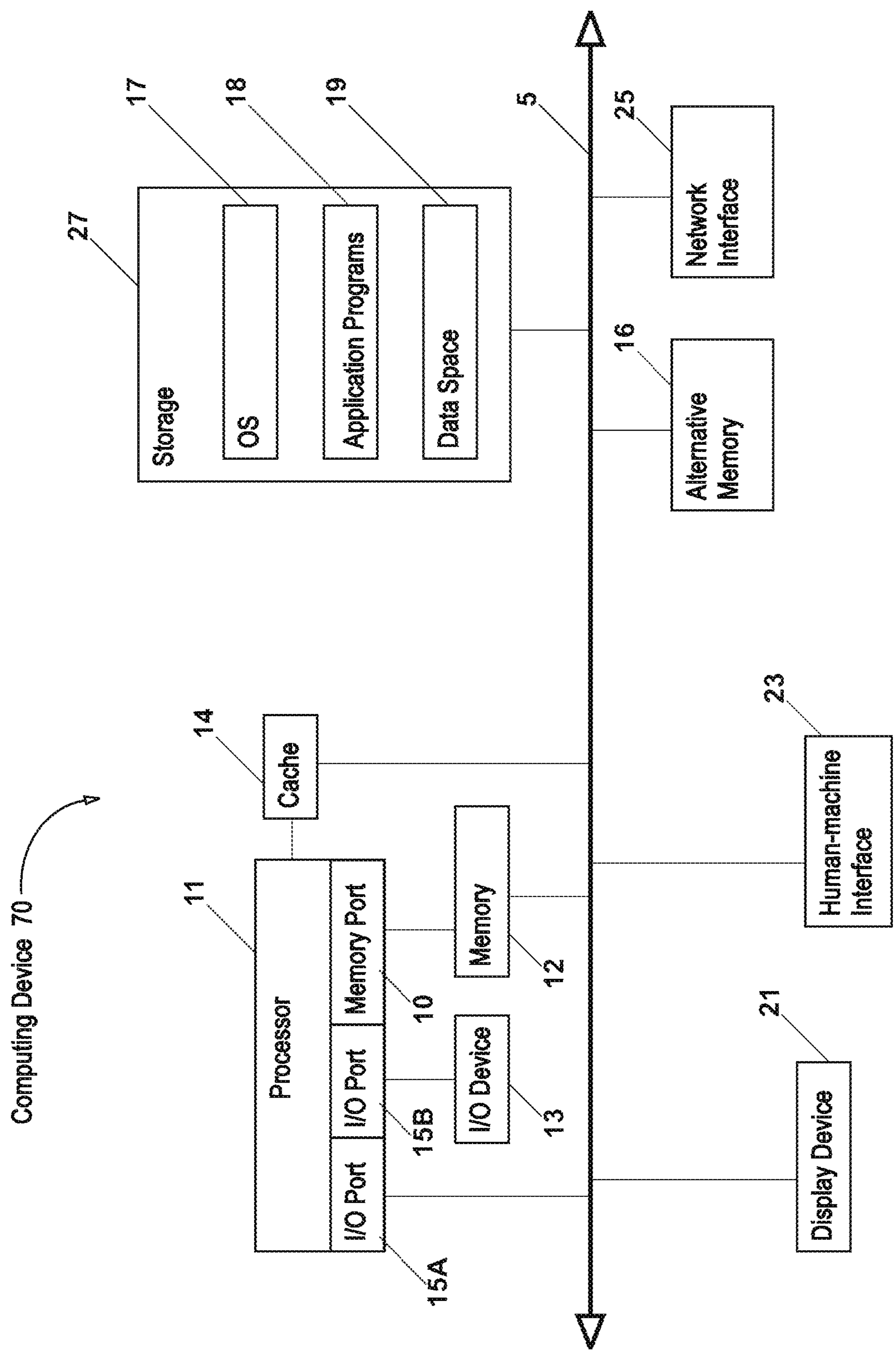

Computing Device 70
Storage
OS
Application Programs
Data Space
Network Interface
Alternative Memory
Cache
Processor
Memory Port
I/O Port
I/O Port
Memory
I/O Device
Human-machine Interface
Display Device
27
17
18
19
5
25
16
14
11
10
12
15B
13
15A
23
21

FIG. 1

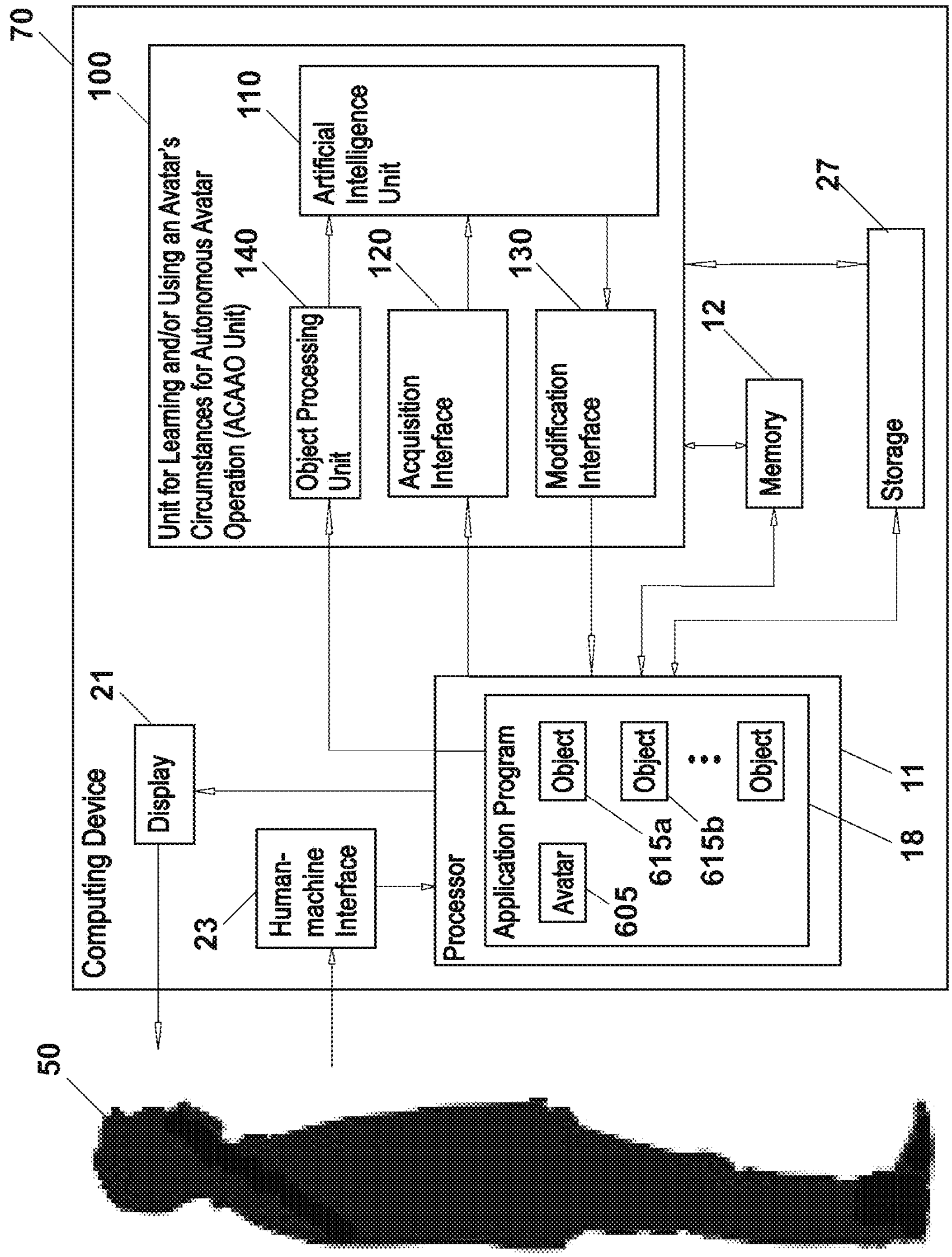

70
Computing Device
100
Unit for Learning and/or Using an Avatar's Circumstances for Autonomous Avatar Operation (ACAAO Unit)
110
Artificial Intelligence Unit
140
Object Processing Unit
120
Acquisition Interface
130
Modification Interface
12
Memory
27
Storage
21
Display
23
Human-machine Interface
Processor
Application Program
Avatar
605
Object
615a
Object
615b
Object
11
18
50

FIG. 2

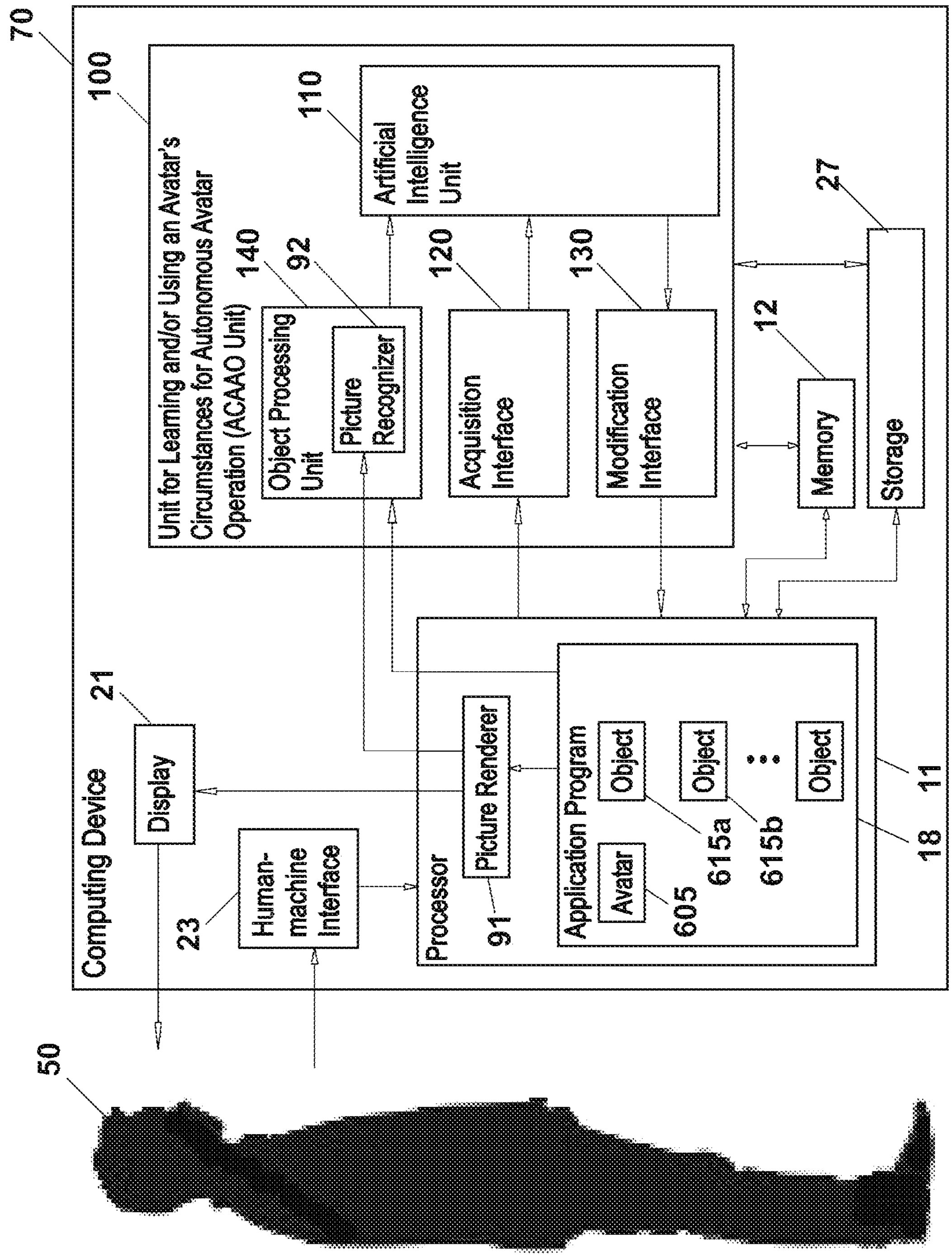

70
100
Unit for Learning and/or Using an Avatar's Circumstances for Autonomous Avatar Operation (ACAAO Unit)
110
Artificial Intelligence Unit
140
92
Object Processing Unit
Picture Recognizer
120
Acquisition Interface
130
Modification Interface
27
Storage
12
Memory
Computing Device
21
Display
23
Human-machine Interface
Processor
91
Picture Renderer
Application Program
Avatar
605
Object
615a
Object
615b
Object
11
18
50

FIG. 3

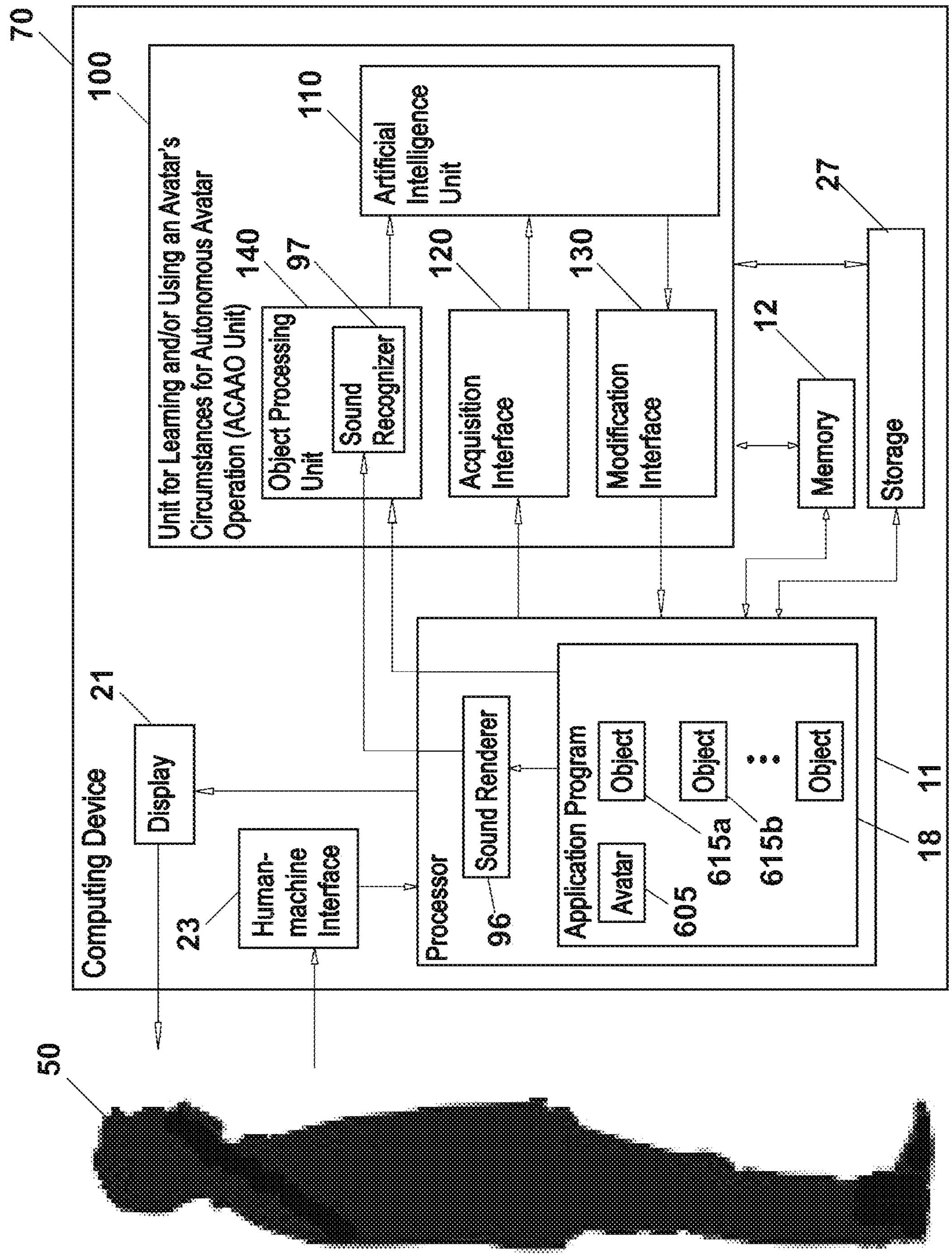

70
Computing Device
100
Unit for Learning and/or Using an Avatar's Circumstances for Autonomous Avatar Operation (ACAAO Unit)
110
Artificial Intelligence Unit
140
97
Object Processing Unit
Sound Recognizer
120
Acquisition Interface
130
Modification Interface
12
Memory
27
Storage
21
Display
23
Human-machine Interface
Processor
96
Sound Renderer
Application Program
Avatar
605
Object
615a
Object
615b
Object
18
11
50

FIG. 4

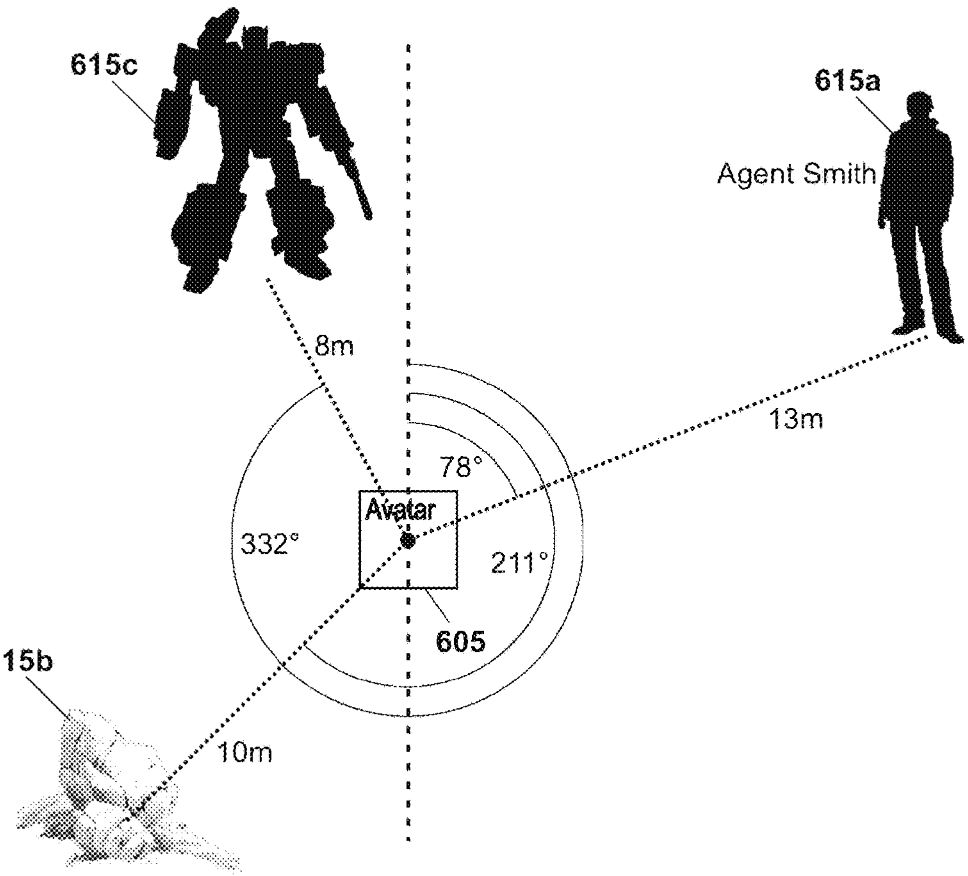

615c
615a
Agent Smith
8m
13m
78°
Avatar
332°
211°
605
615b
10m

FIG. 5A

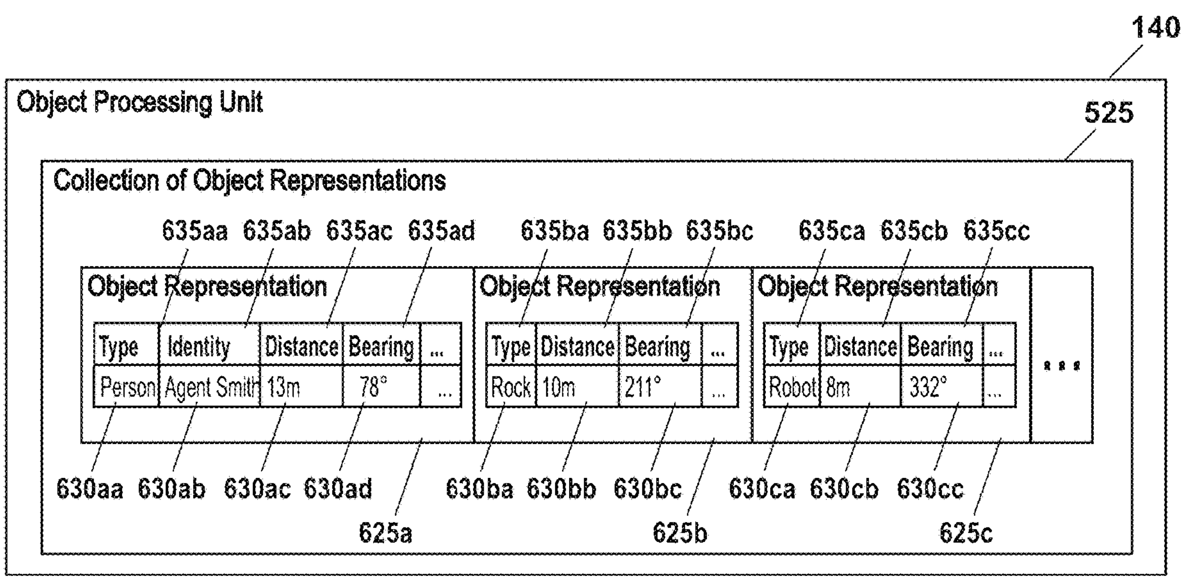

140
Object Processing Unit
525
Collection of Object Representations
635aa 635ab 635ac 635ad
635ba 635bb 635bc
635ca 635cb 635cc
Object Representation
Type | Identity | Distance | Bearing | ...
Person | Agent Smith | 13m | 78° | ...
Object Representation
Type | Distance | Bearing | ...
Rock | 10m | 211° | ...
Object Representation
Type | Distance | Bearing | ...
Robot | 8m | 332° | ...
...
630aa 630ab 630ac 630ad
630ba 630bb 630bc
630ca 630cb 630cc
625a
625b
625c

FIG. 5B

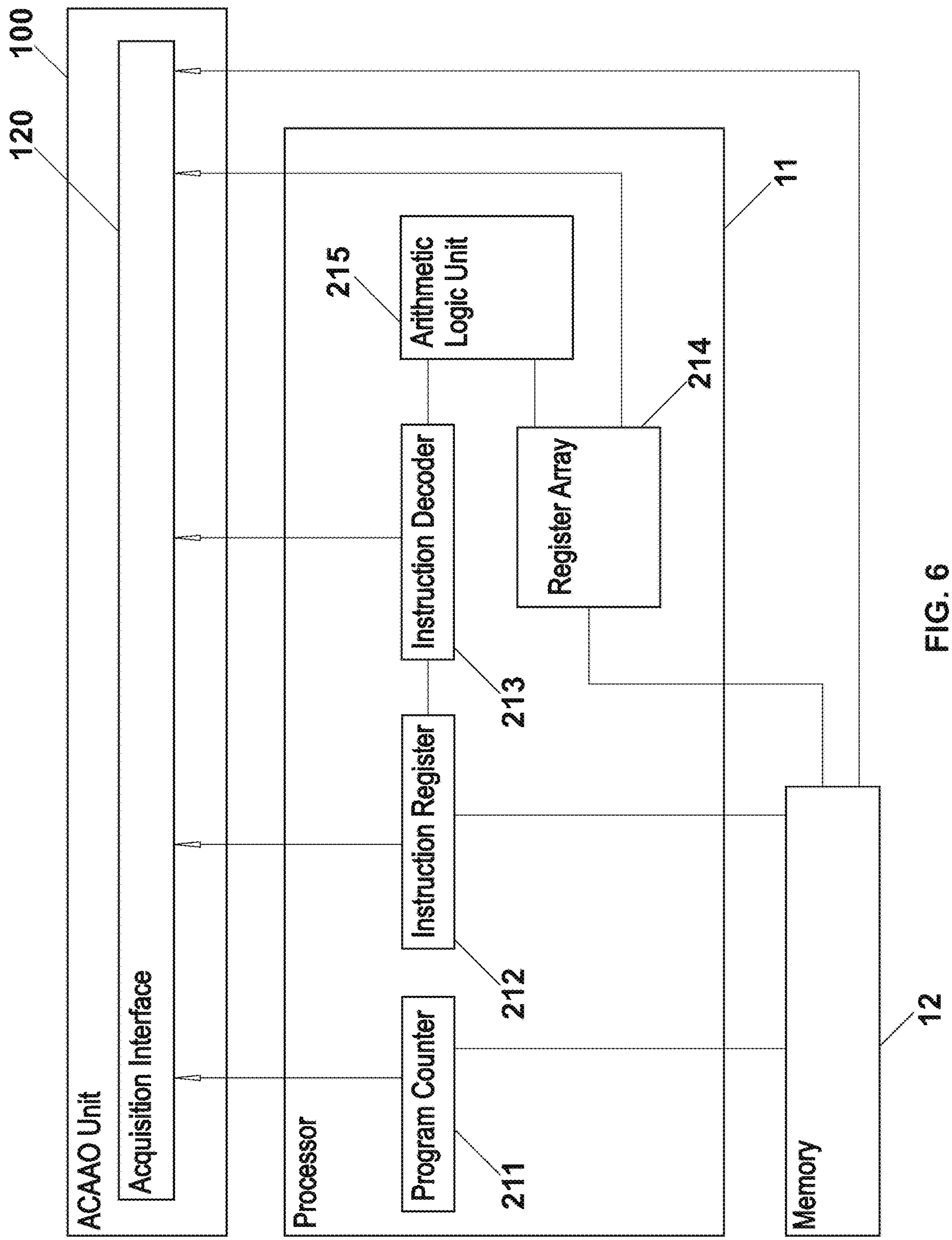

100
120
ACAAO Unit
Acquisition Interface
Processor
11
215
Arithmetic
Logic Unit
214
Instruction Decoder
Register Array
213
Instruction Register
212
Program Counter
211
Memory
12

FIG. 6

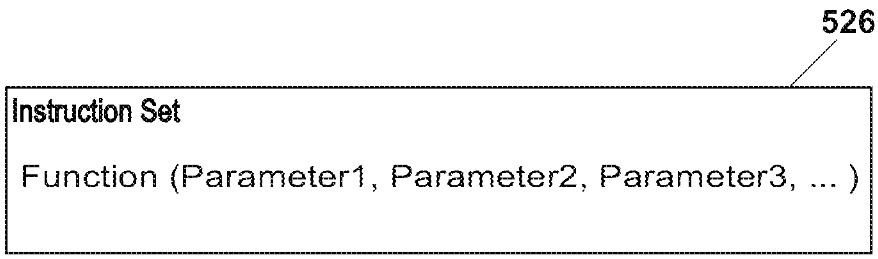


FIG. 7A

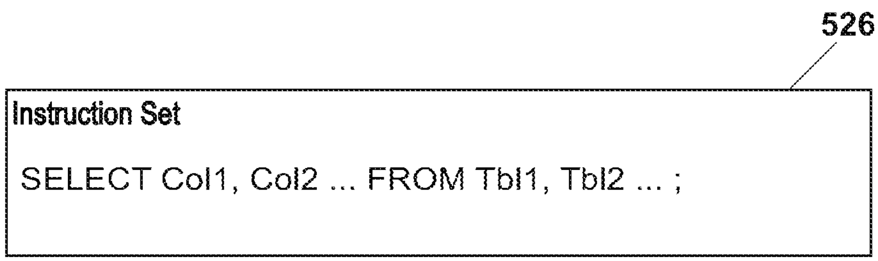


FIG. 7B

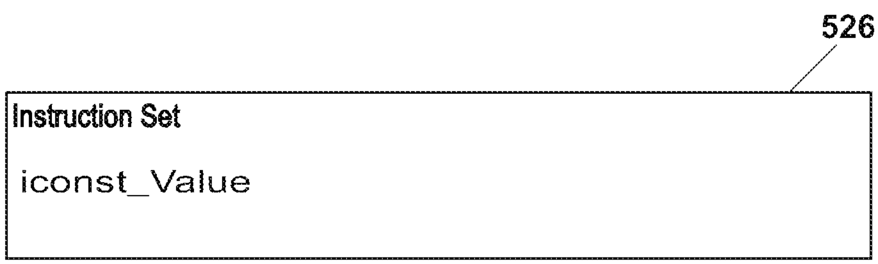


FIG. 7C

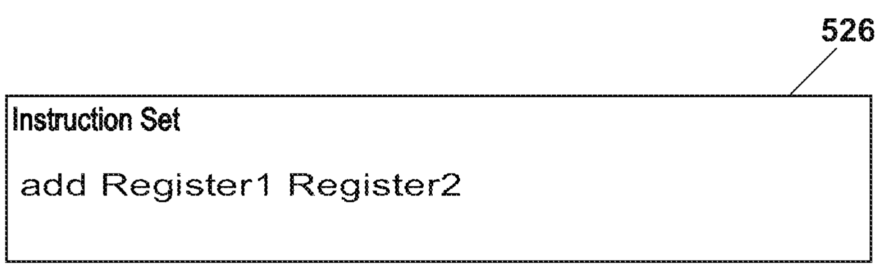


FIG. 7D

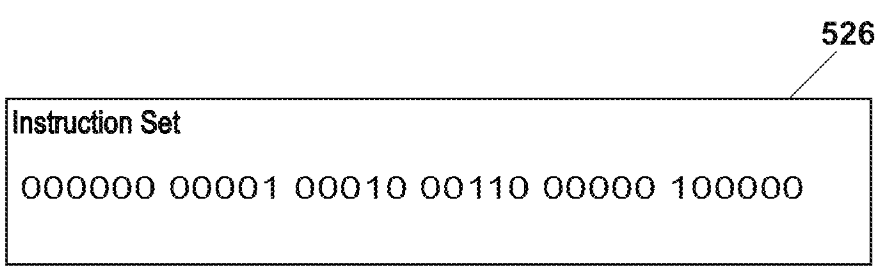


FIG. 7E

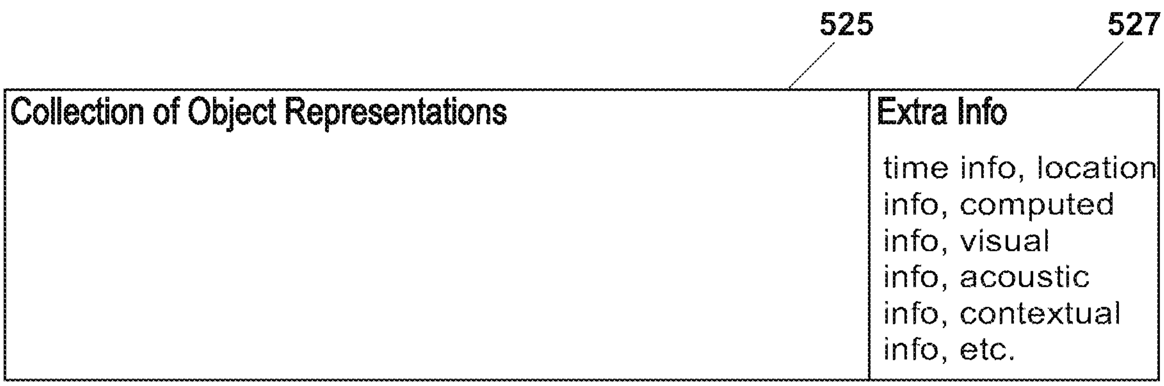

525
527
Collection of Object Representations
Extra Info
time info, location info, computed info, visual info, acoustic info, contextual info, etc.

FIG. 8A

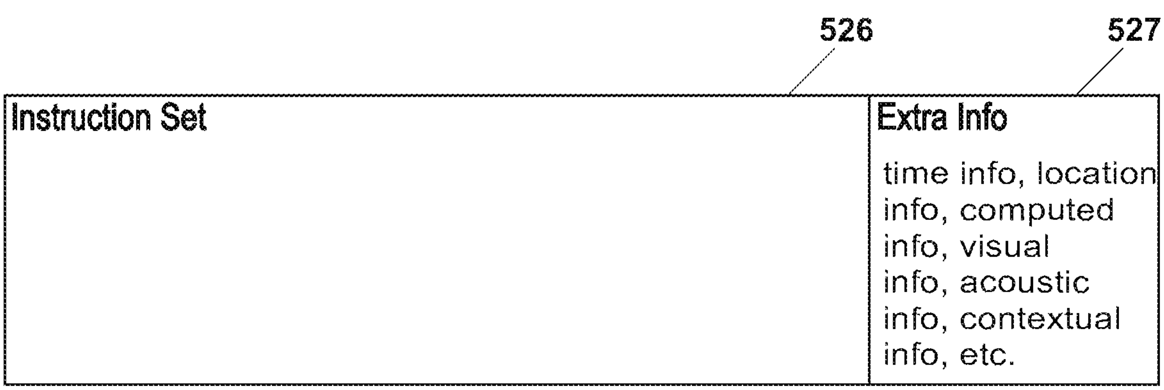

526
527
Instruction Set
Extra Info
time info, location info, computed info, visual info, acoustic info, contextual info, etc.

FIG. 8B

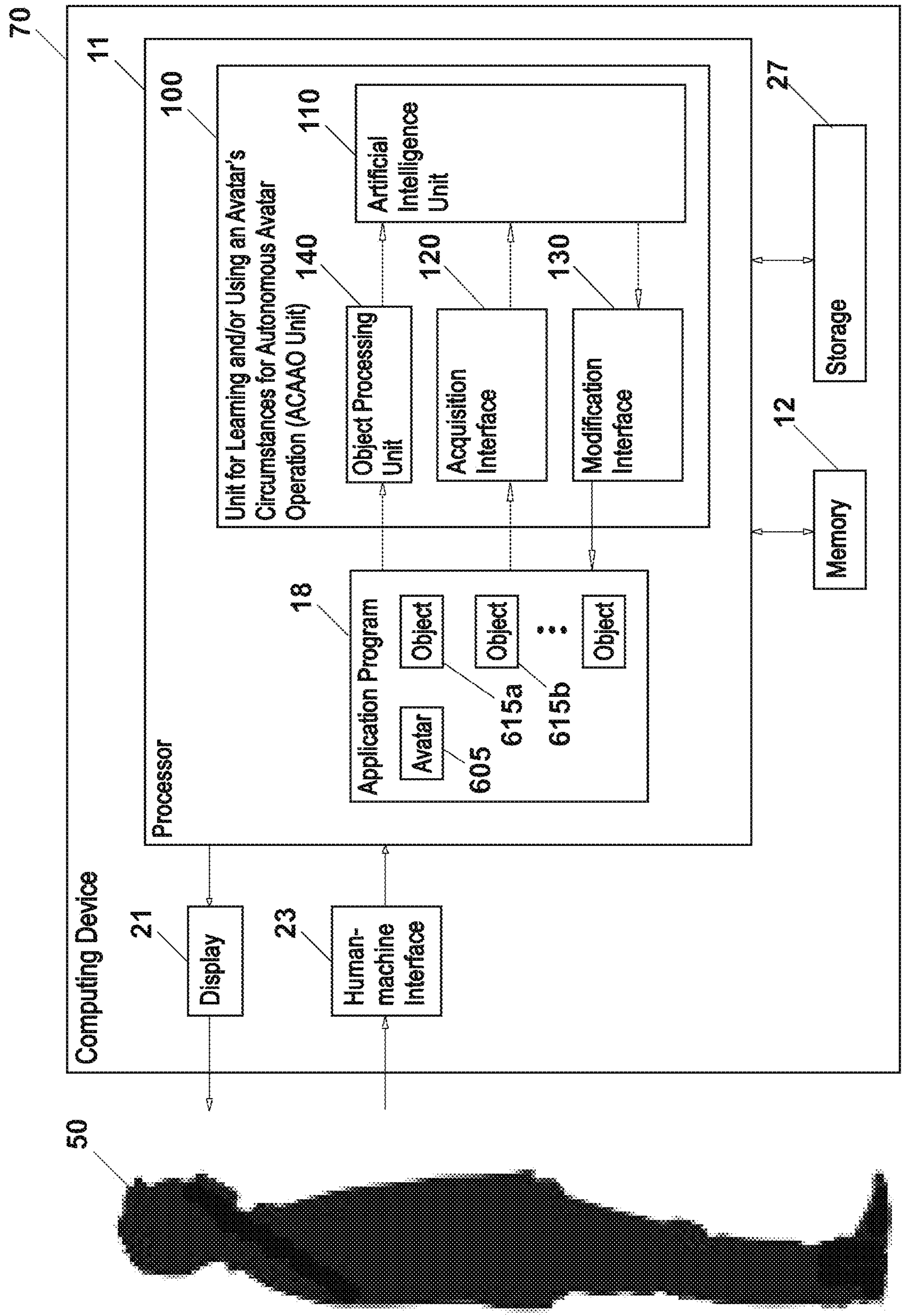

70
Computing Device
11
Processor
100
Unit for Learning and/or Using an Avatar's Circumstances for Autonomous Avatar Operation (ACAAO Unit)
110
Artificial Intelligence Unit
140
Object Processing Unit
120
Acquisition Interface
130
Modification Interface
18
Application Program
Avatar
605
Object
615a
Object
615b
Object
27
Storage
12
Memory
21
Display
23
Human-machine Interface
50

FIG. 9

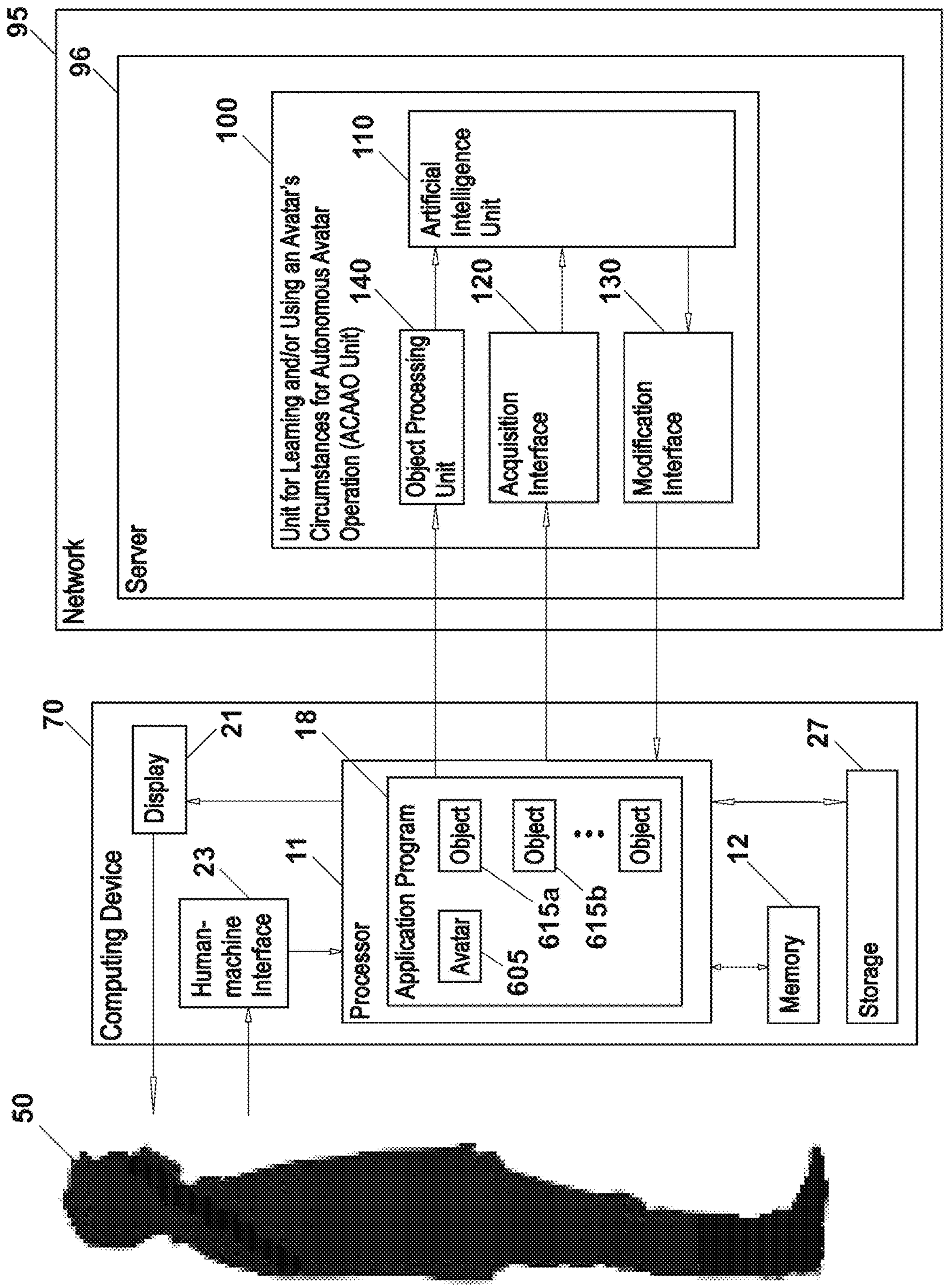
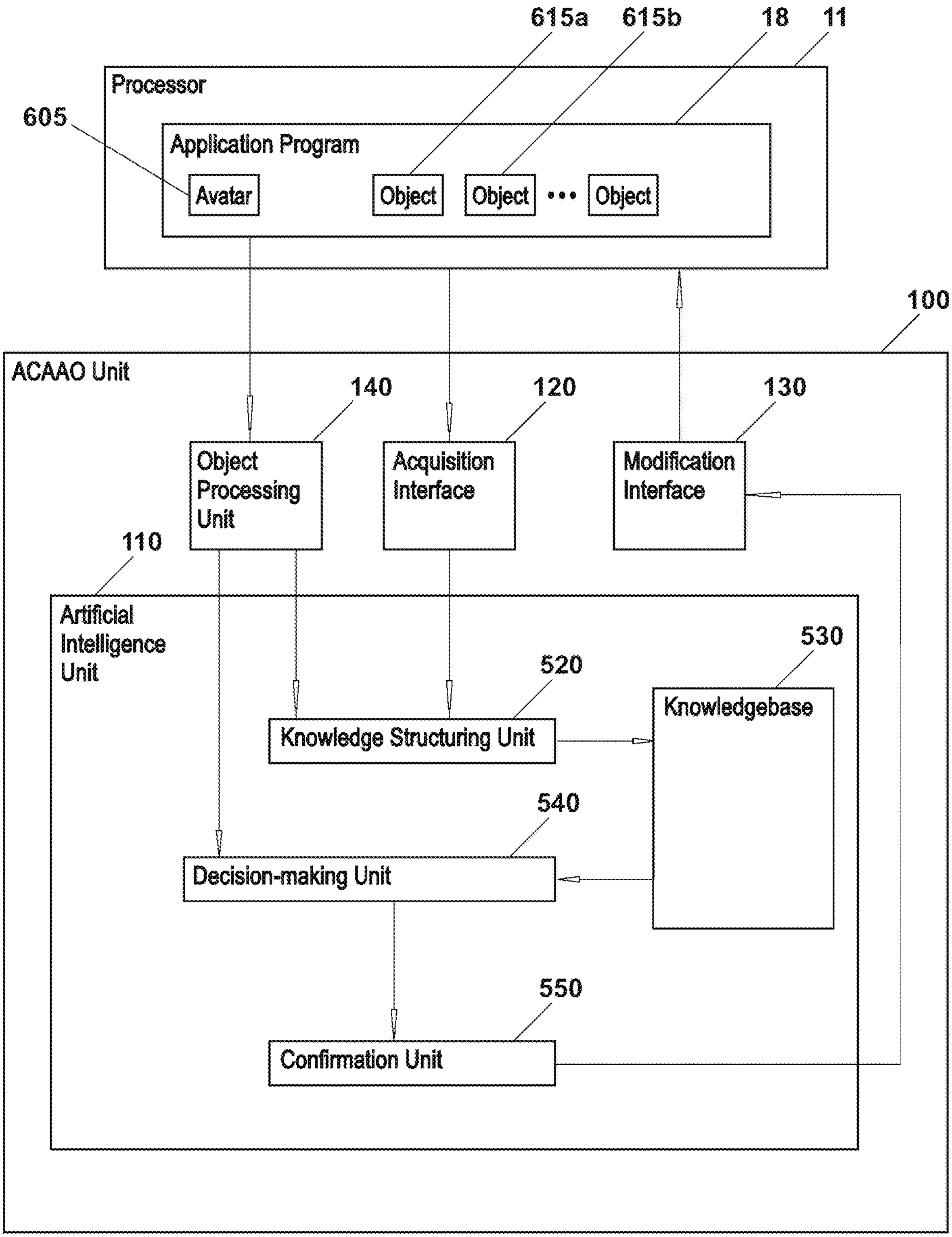
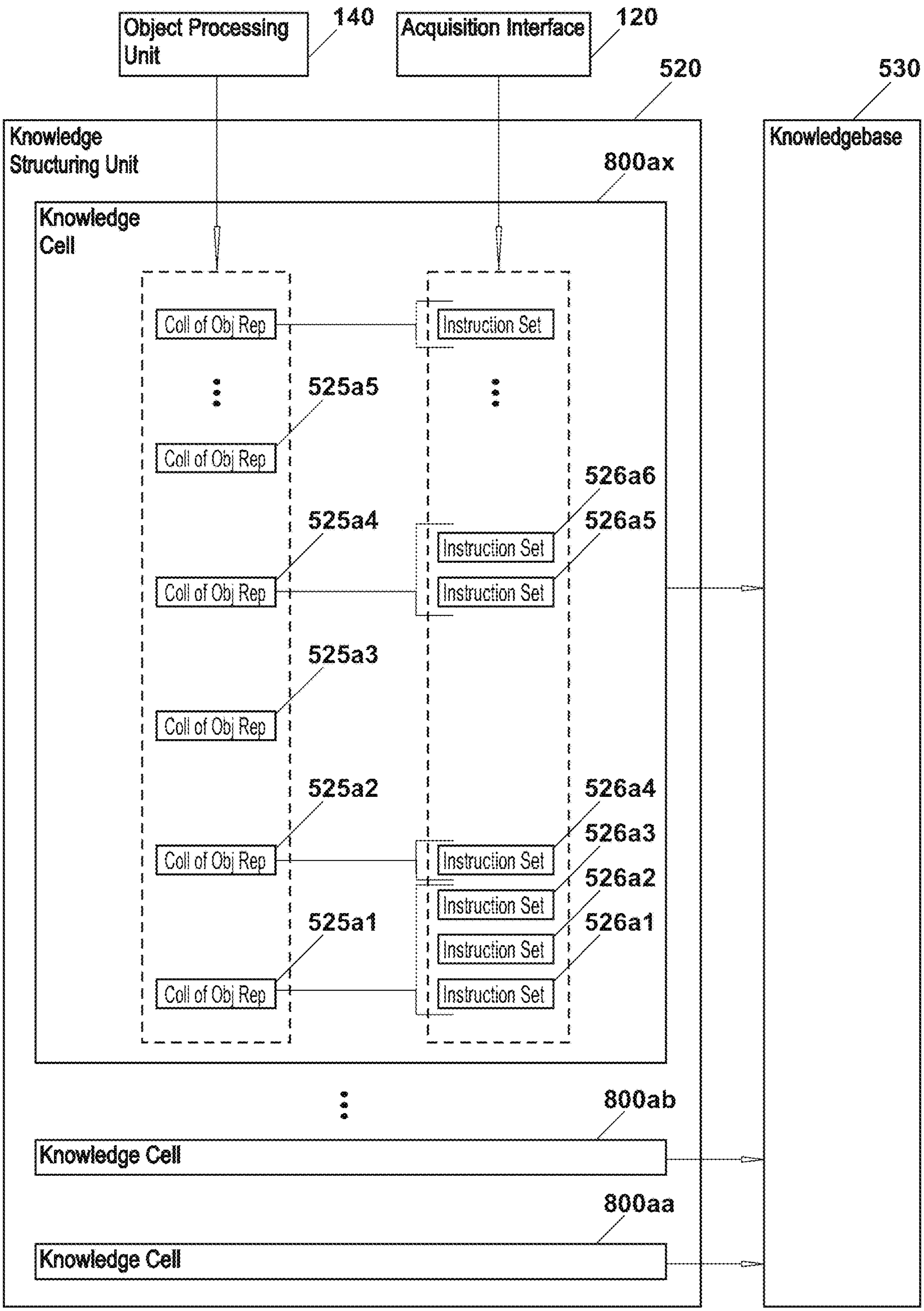

95
Network
96
Server
100
Unit for Learning and/or Using an Avatar's Circumstances for Autonomous Avatar Operation (ACAAO Unit)
110
Artificial Intelligence Unit
140
Object Processing Unit
120
Acquisition Interface
130
Modification Interface
70
Computing Device
21
Display
23
Human-machine Interface
11
Processor
18
Application Program
Avatar
605
Object
615a
Object
615b
Object
12
Memory
27
Storage
50

FIG. 10



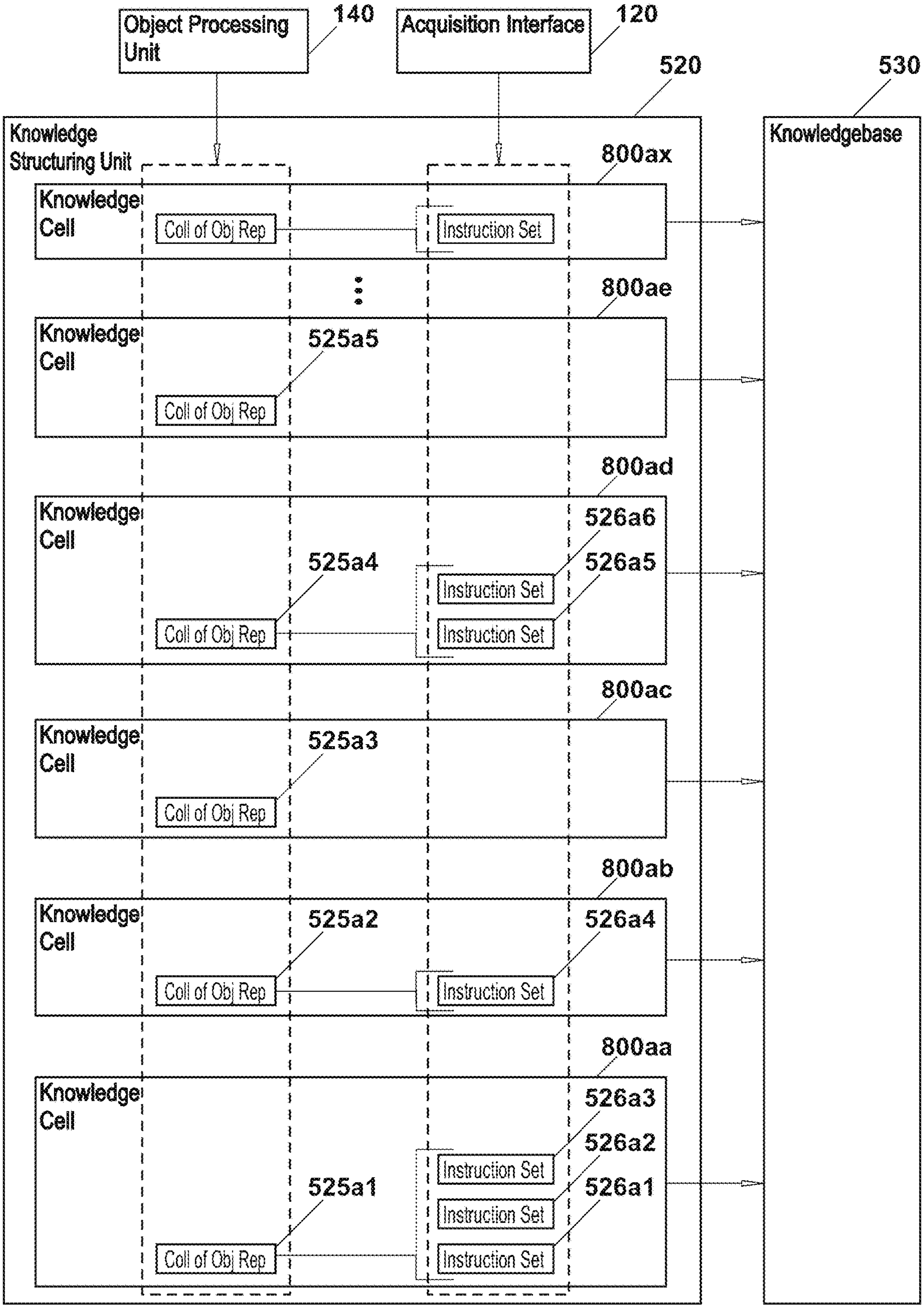
615a
615b
18
11
Processor
605
Application Program
Avatar
Object
Object
Object
100
ACAAO Unit
140
120
130
Object Processing Unit
Acquisition Interface
Modification Interface
110
Artificial Intelligence Unit
530
520
Knowledge Structuring Unit
Knowledgebase
540
Decision-making Unit
550
Confirmation Unit

FIG. 11



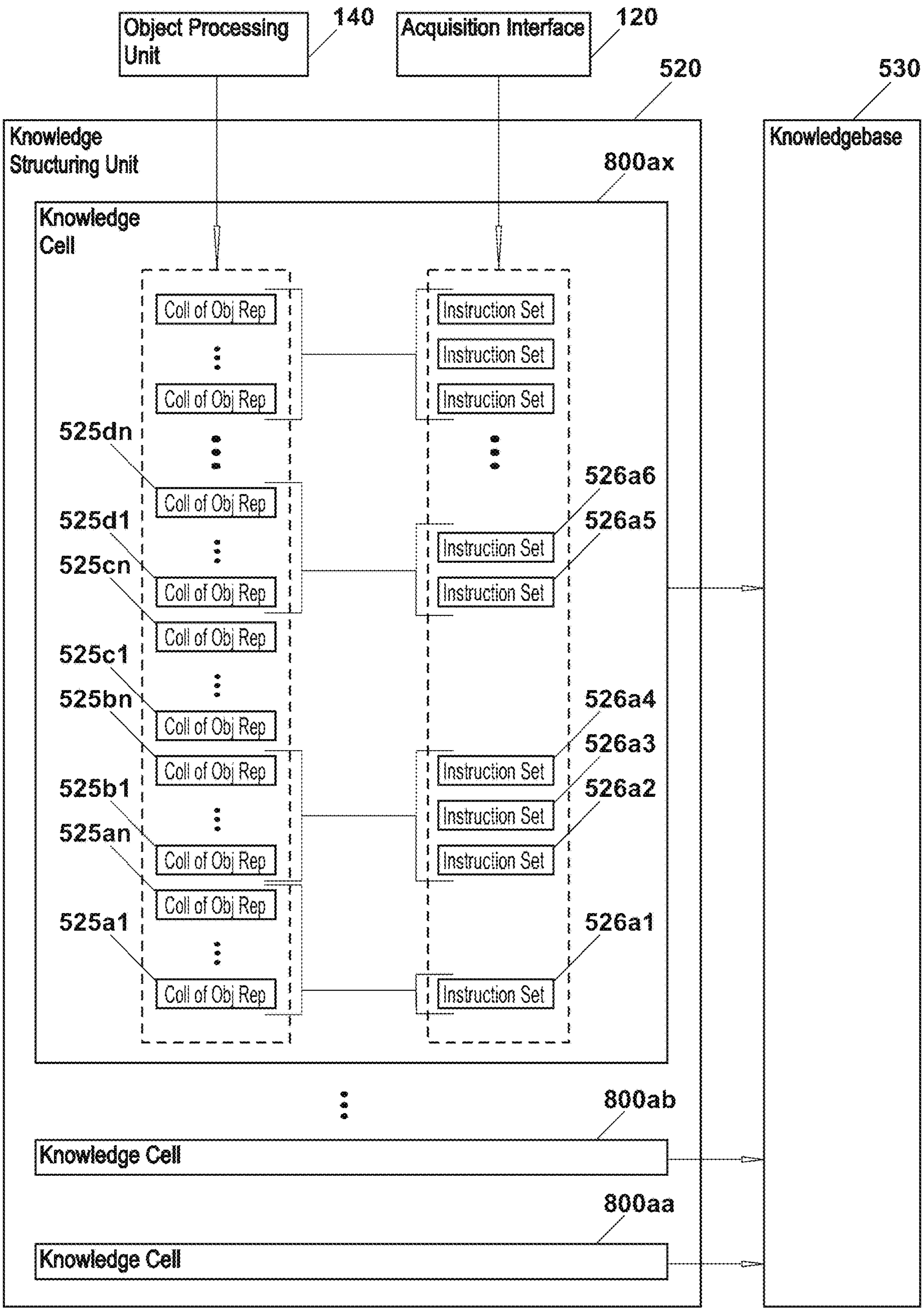
Object Processing Unit
140
Acquisition Interface
120
520
530
Knowledge Structuring Unit
800ax
Knowledgebase
Knowledge Cell
Coll of Obj Rep
Instruction Set
525a5
Coll of Obj Rep
526a6
525a4
526a5
Instruction Set
Coll of Obj Rep
Instruction Set
525a3
Coll of Obj Rep
525a2
526a4
526a3
Coll of Obj Rep
Instruction Set
526a2
Instruction Set
525a1
526a1
Instruction Set
Coll of Obj Rep
Instruction Set
800ab
Knowledge Cell
800aa
Knowledge Cell

FIG. 12



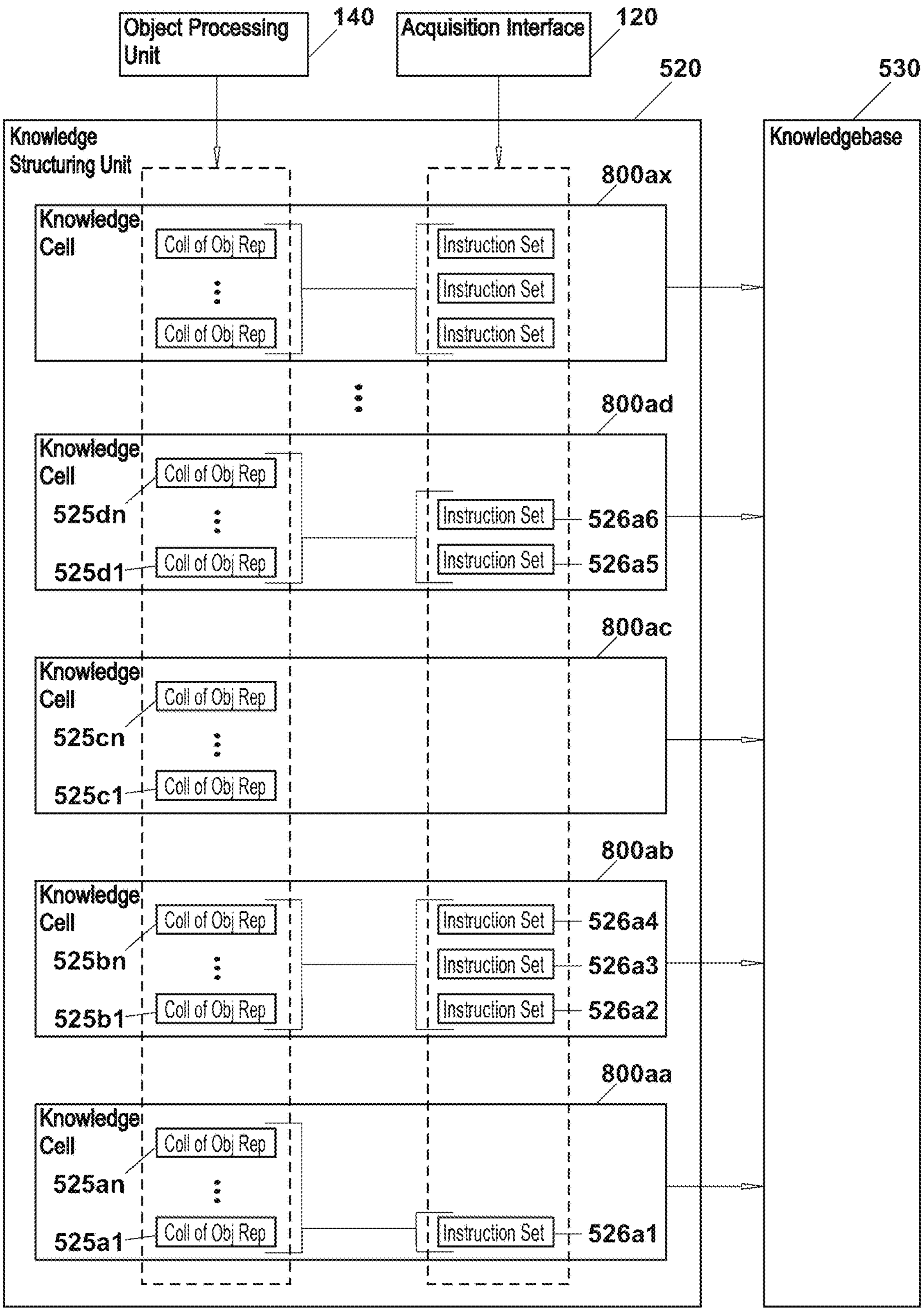
Object Processing Unit
140
Acquisition Interface
120
520
530
Knowledge Structuring Unit
Knowledgebase
800ax
Knowledge Cell
Coll of Obj Rep
Instruction Set
800ae
Knowledge Cell
525a5
Coll of Obj Rep
800ad
Knowledge Cell
526a6
525a4
526a5
Instruction Set
Coll of Obj Rep
Instruction Set
800ac
Knowledge Cell
525a3
Coll of Obj Rep
800ab
Knowledge Cell
525a2
526a4
Coll of Obj Rep
Instruction Set
800aa
Knowledge Cell
526a3
526a2
525a1
Instruction Set
526a1
Instruction Set
Coll of Obj Rep
Instruction Set

FIG. 13



Object Processing Unit
140
Acquisition Interface
120
520
530
Knowledge Structuring Unit
800ax
Knowledgebase
Knowledge Cell
Coll of Obj Rep
Instruction Set
525dn
525d1
525cn
525c1
525bn
525b1
525an
525a1
526a6
526a5
526a4
526a3
526a2
526a1
800ab
Knowledge Cell
800aa
Knowledge Cell

FIG. 14



Object Processing Unit
140
Acquisition Interface
120
520
530
Knowledge Structuring Unit
Knowledgebase
800ax
Knowledge Cell
Coll of Obj Rep
Instruction Set
800ad
525dn
525d1
526a6
526a5
800ac
525cn
525c1
800ab
525bn
525b1
526a4
526a3
526a2
800aa
525an
525a1
526a1

FIG. 15

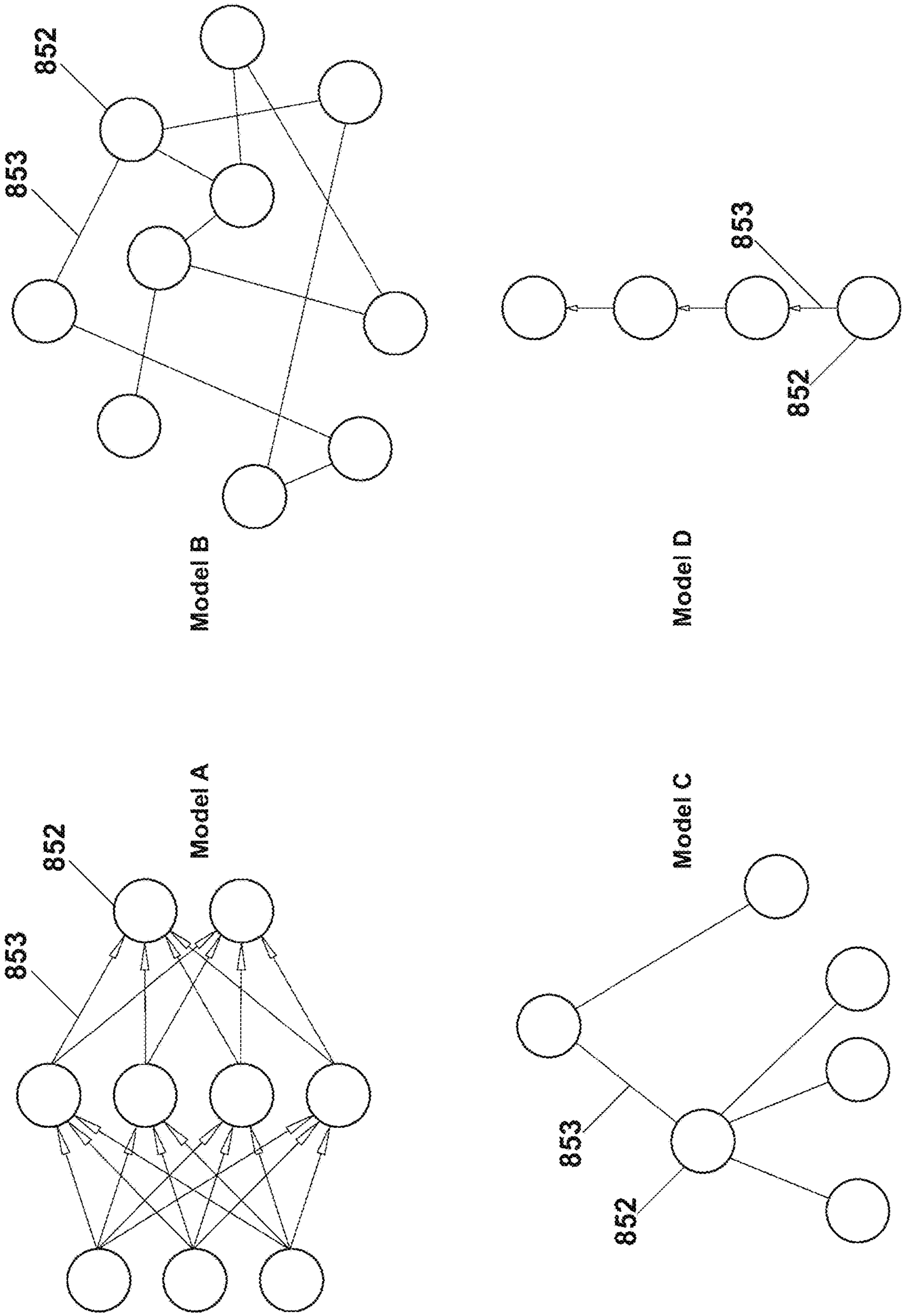


FIG. 16

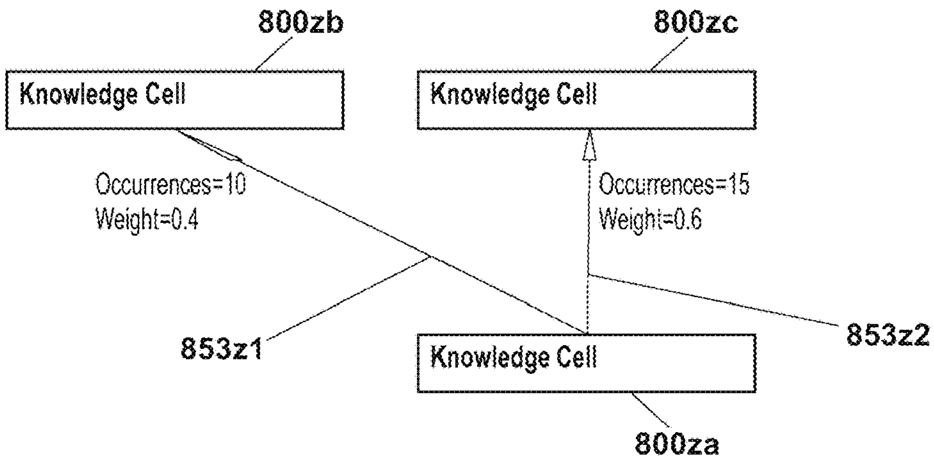


FIG. 17A

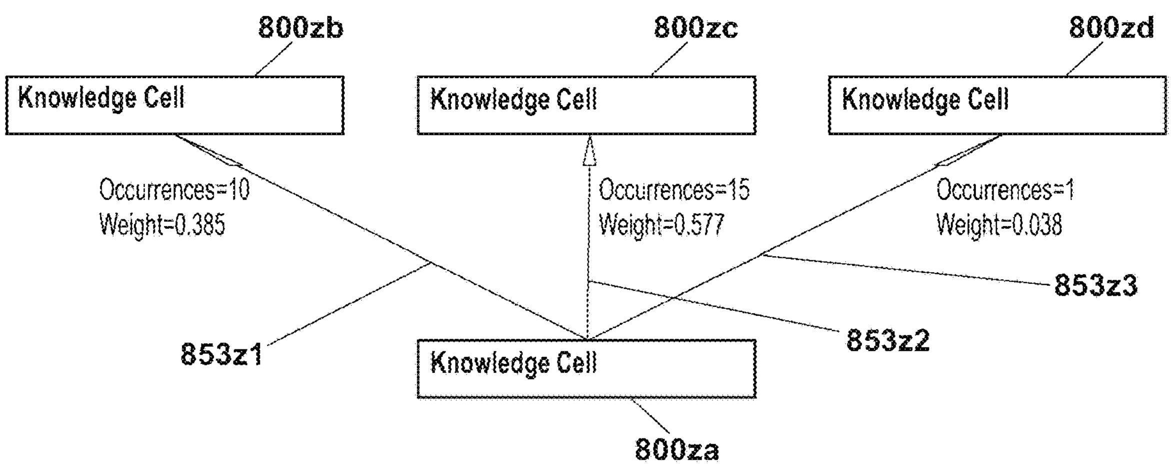


FIG. 17B

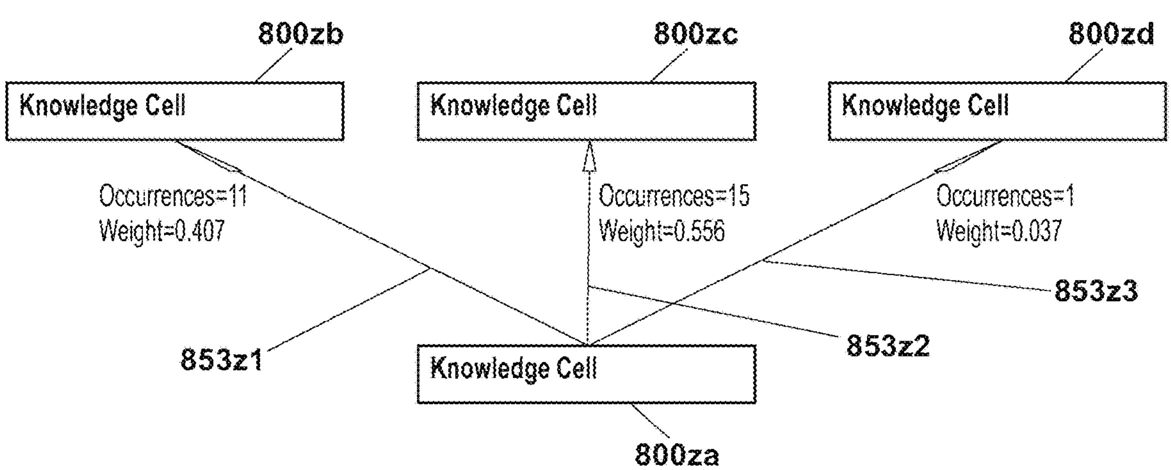


FIG. 17C

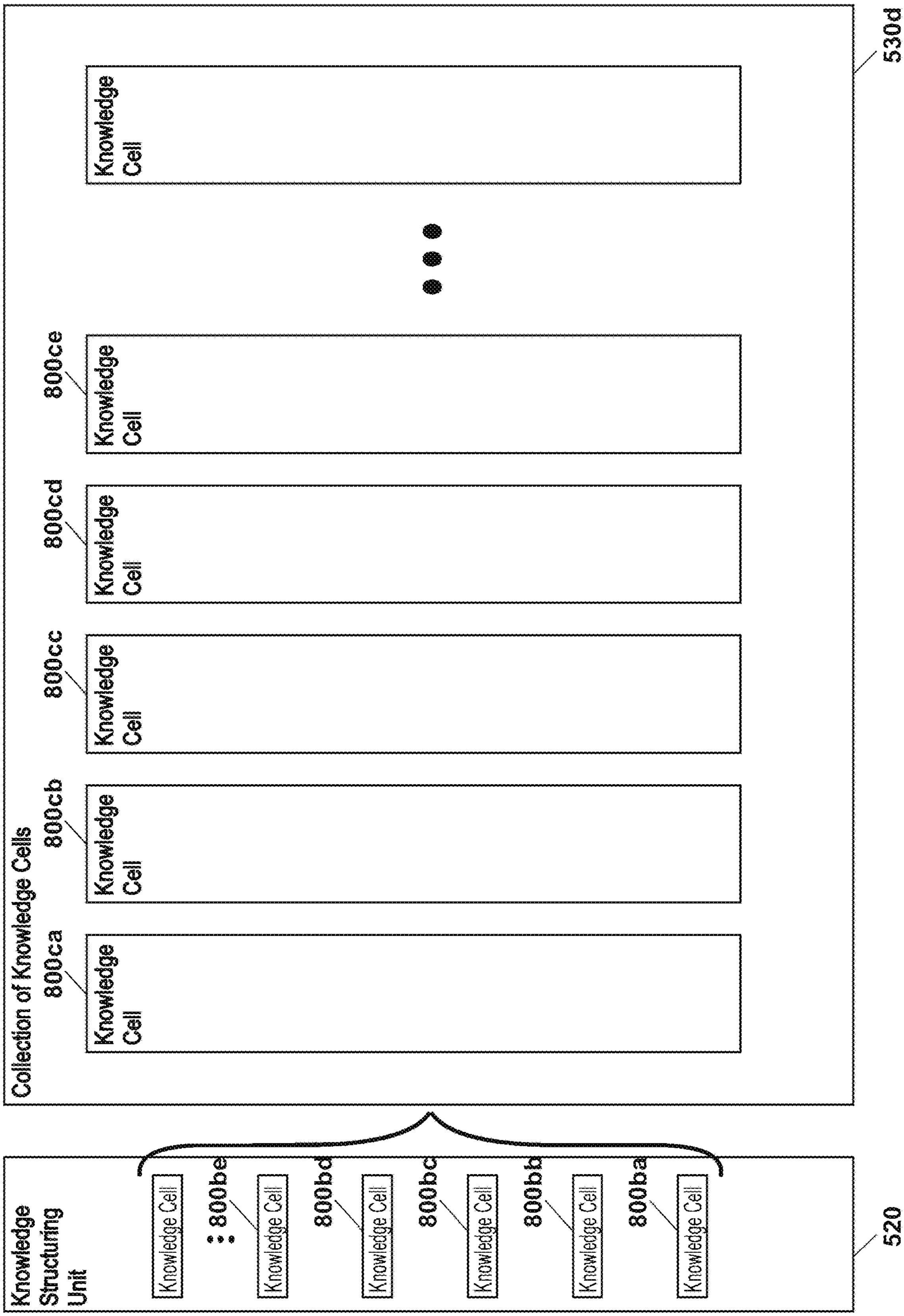

Collection of Knowledge Cells
Knowledge Cell
800ca
800cb
800cc
800cd
800ce
530d
Knowledge Structuring Unit
800ba
800bb
800bc
800bd
800be
520

FIG. 18

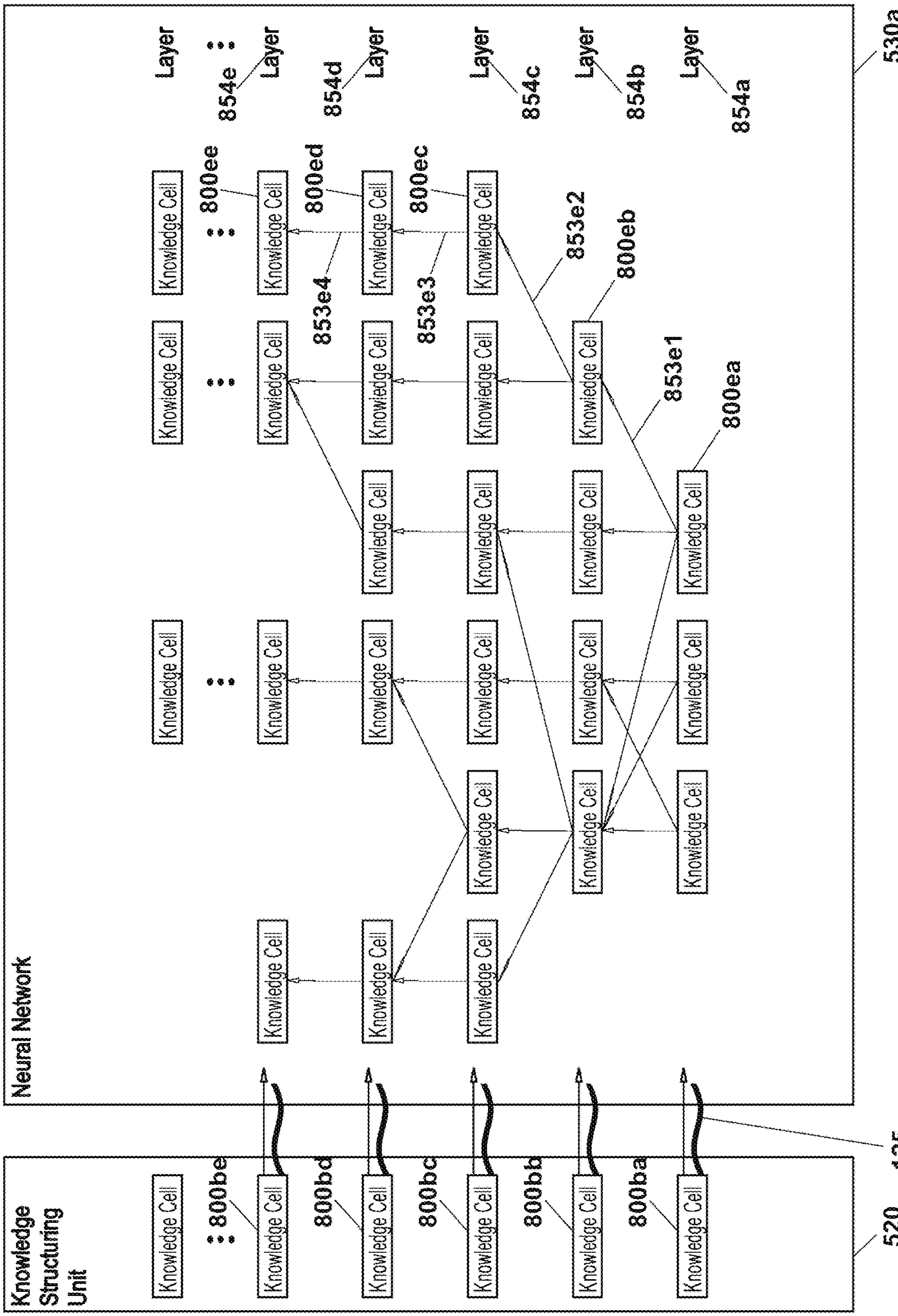
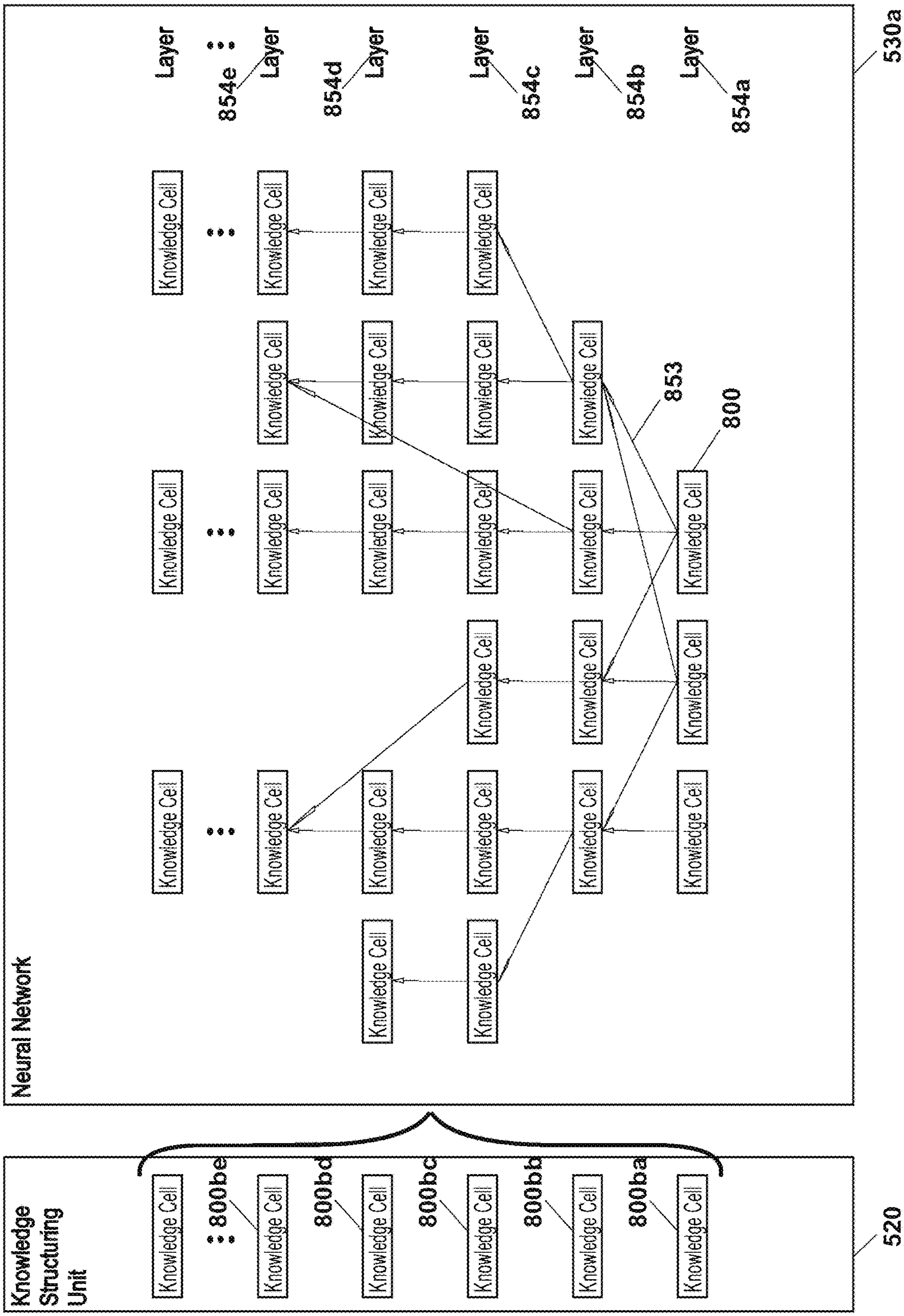

Neural Network
Knowledge Structuring Unit
Knowledge Cell
Layer
530a
520
125
800ba
800bb
800bc
800bd
800be
800ea
800eb
800ec
800ed
800ee
853e1
853e2
853e3
853e4
854a
854b
854c
854d
854e

FIG. 19



Neural Network
Layer
Knowledge Cell
854a
854b
854c
854d
854e
853
800
530a
Knowledge Structuring Unit
800ba
800bb
800bc
800bd
800be
520

FIG. 20

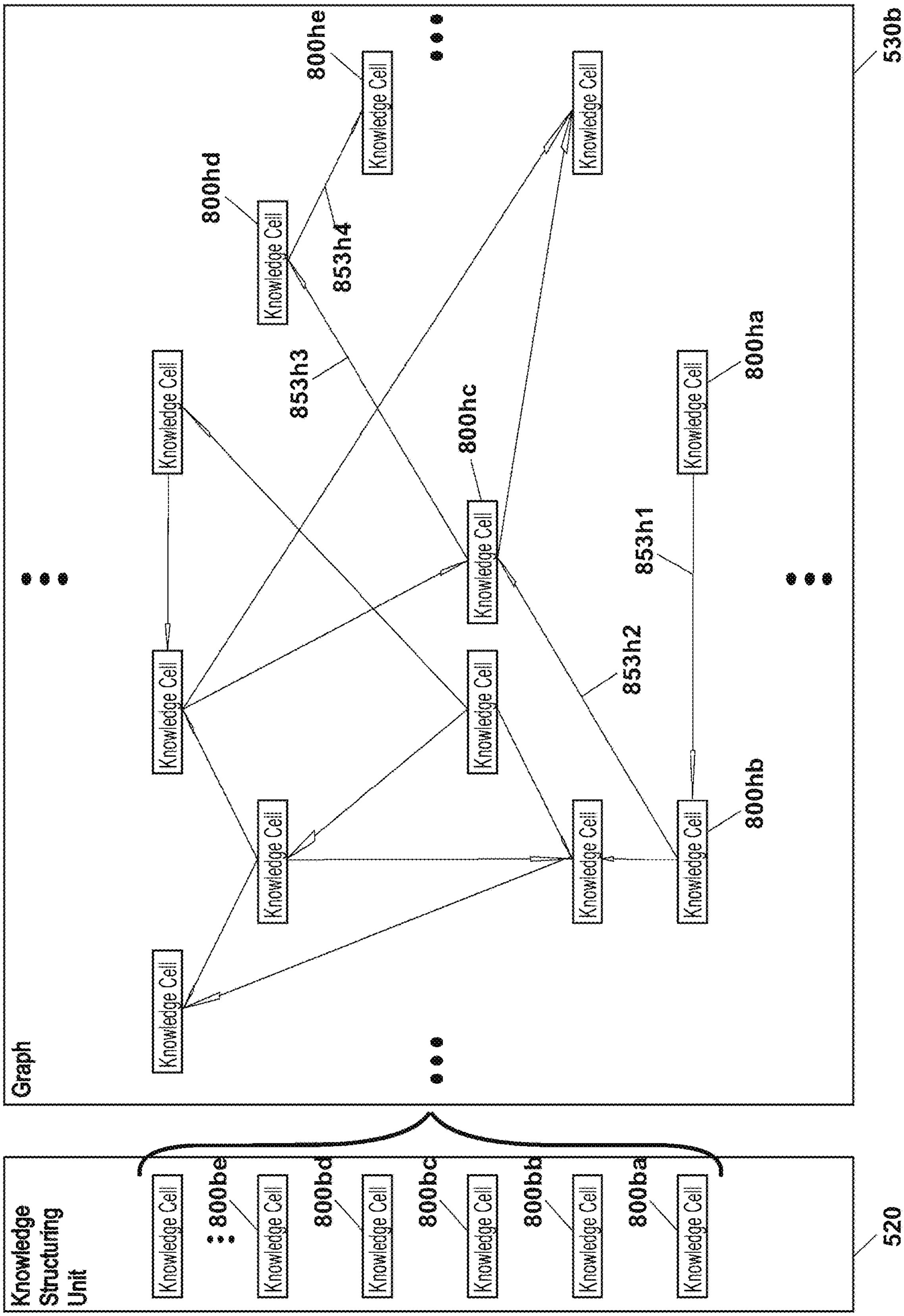

Graph
Knowledge Cell
800he
800hd
853h4
853h3
800hc
800ha
853h1
853h2
800hb
530b
Knowledge Structuring Unit
800be
800bd
800bc
800bb
800ba
520

FIG. 21

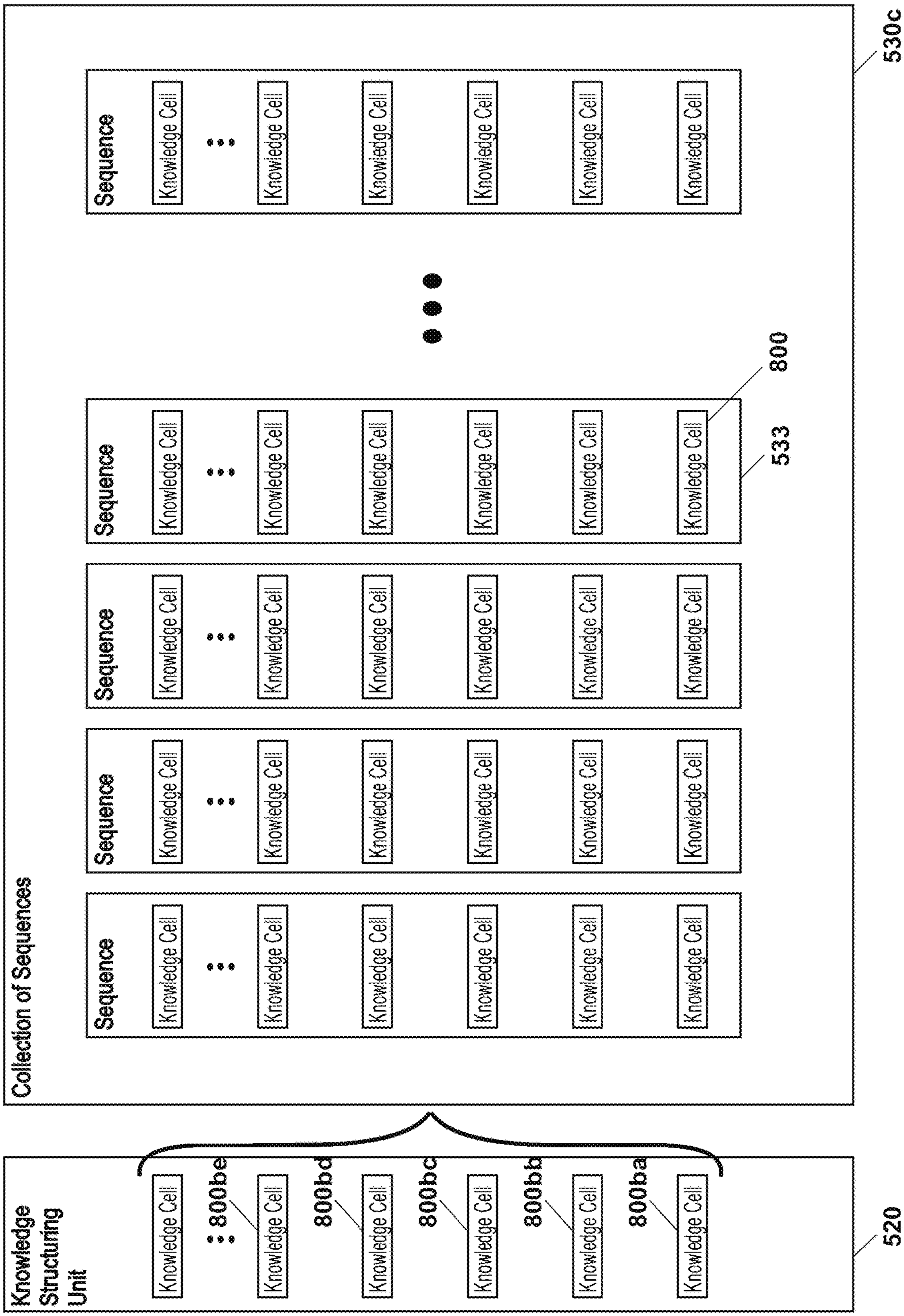

Knowledge Structuring Unit
Knowledge Cell
Knowledge Cell
Knowledge Cell
Knowledge Cell
Knowledge Cell
Knowledge Cell
800be
800bd
800bc
800bb
800ba
520
Collection of Sequences
Sequence
Knowledge Cell
800
533
530c

FIG. 22

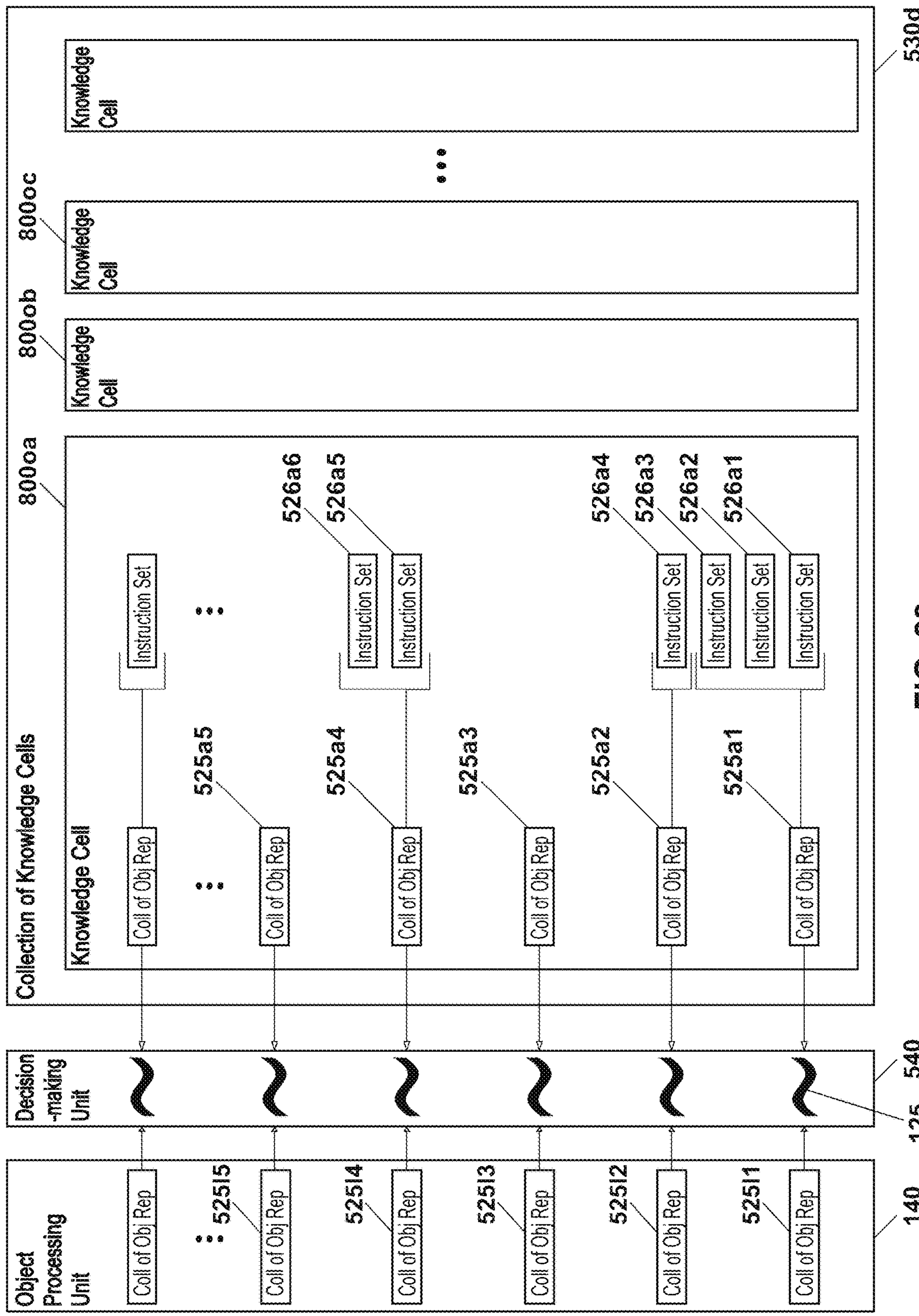

Object Processing Unit
Coll of Obj Rep
525l1
525l2
525l3
525l4
525l5
140
Decision -making Unit
125
540
Collection of Knowledge Cells
Knowledge Cell
Coll of Obj Rep
Instruction Set
525a1
525a2
525a3
525a4
525a5
526a1
526a2
526a3
526a4
526a5
526a6
800oa
800ob
800oc
Knowledge Cell
530d

FIG. 23

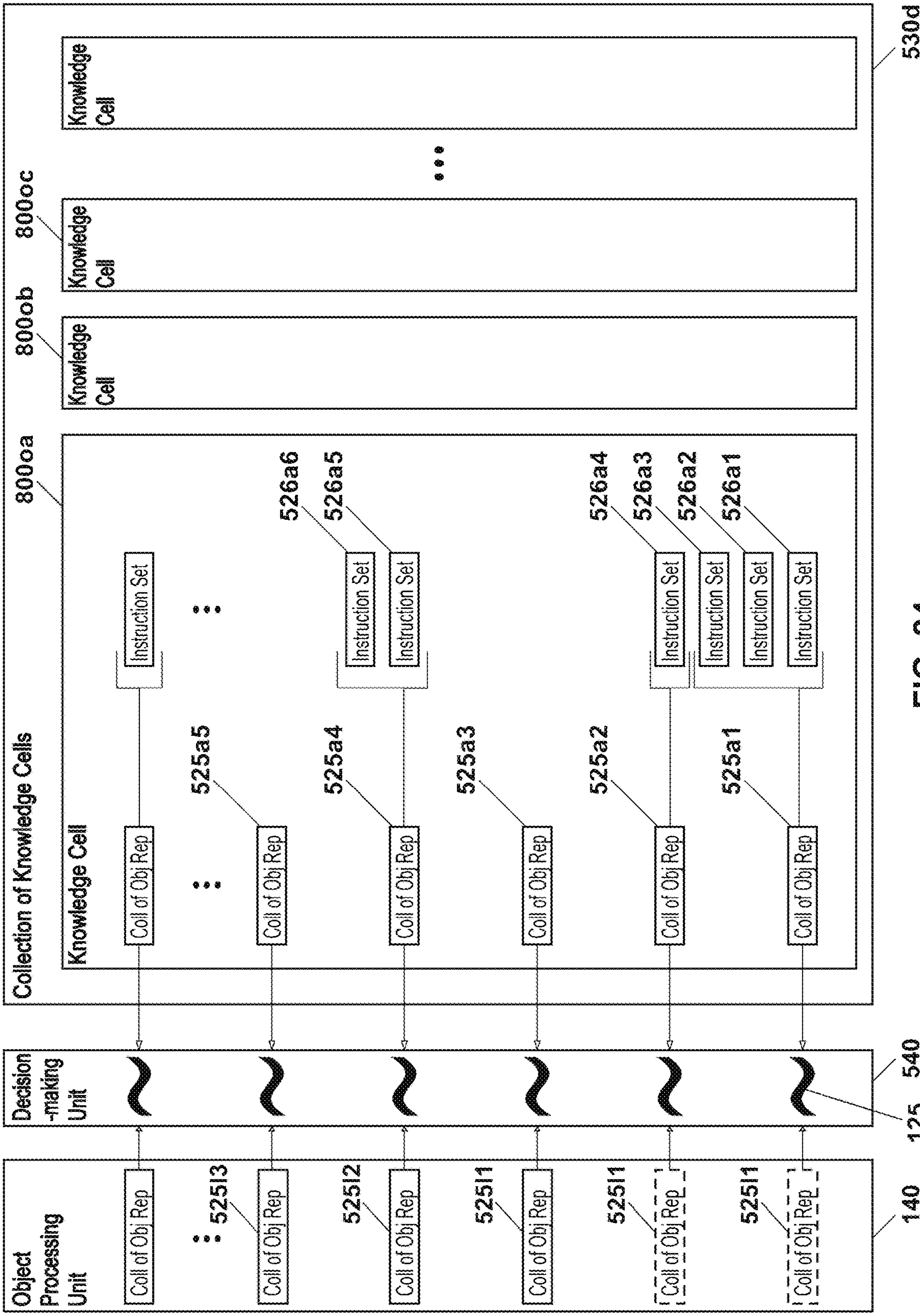

Object Processing Unit
Coll of Obj Rep
525l3
525l2
525l1
Decision -making Unit
125
540
140
Collection of Knowledge Cells
Knowledge Cell
525a1
525a2
525a3
525a4
525a5
Instruction Set
526a1
526a2
526a3
526a4
526a5
526a6
800oa
800ob
800oc
530d

FIG. 24

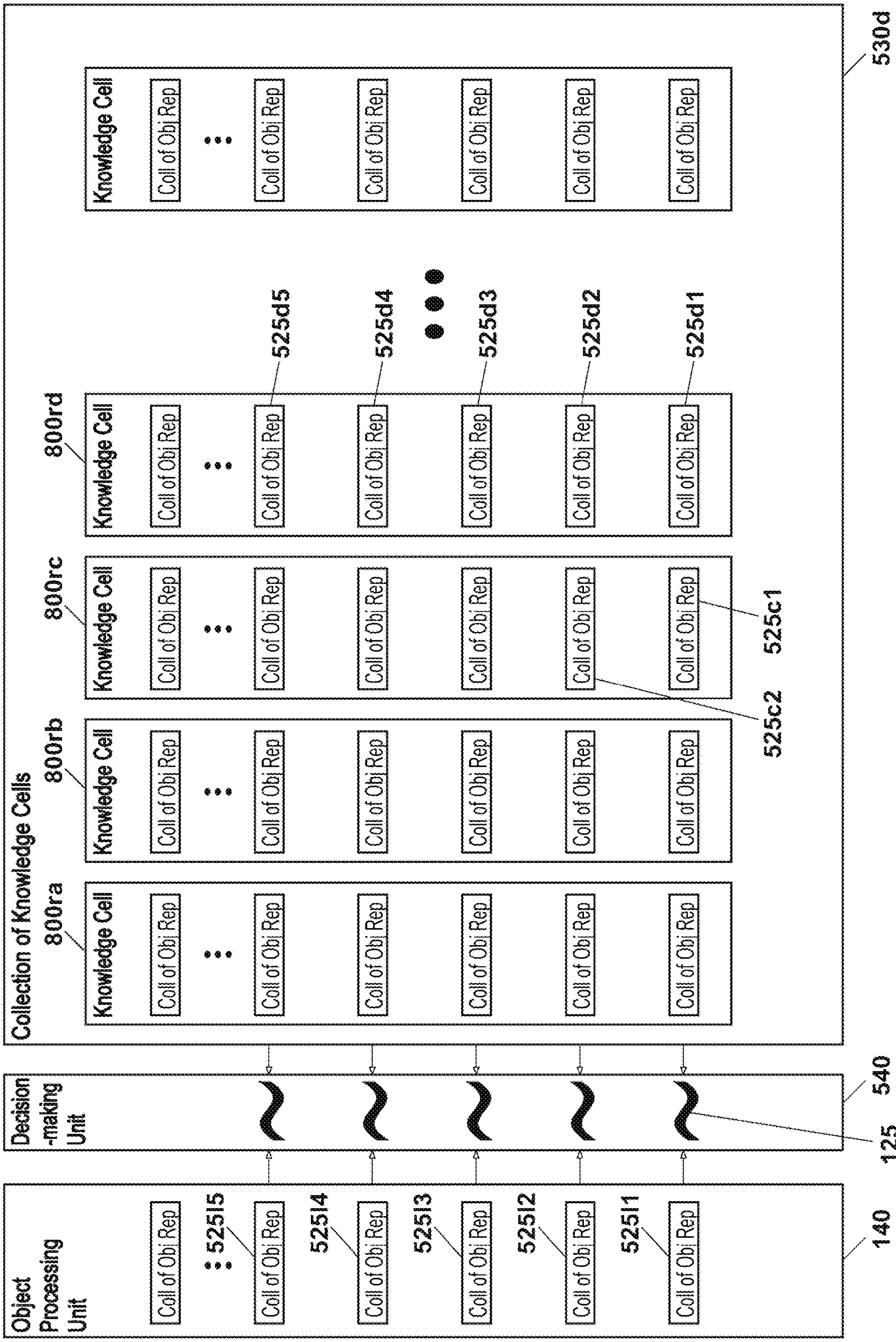
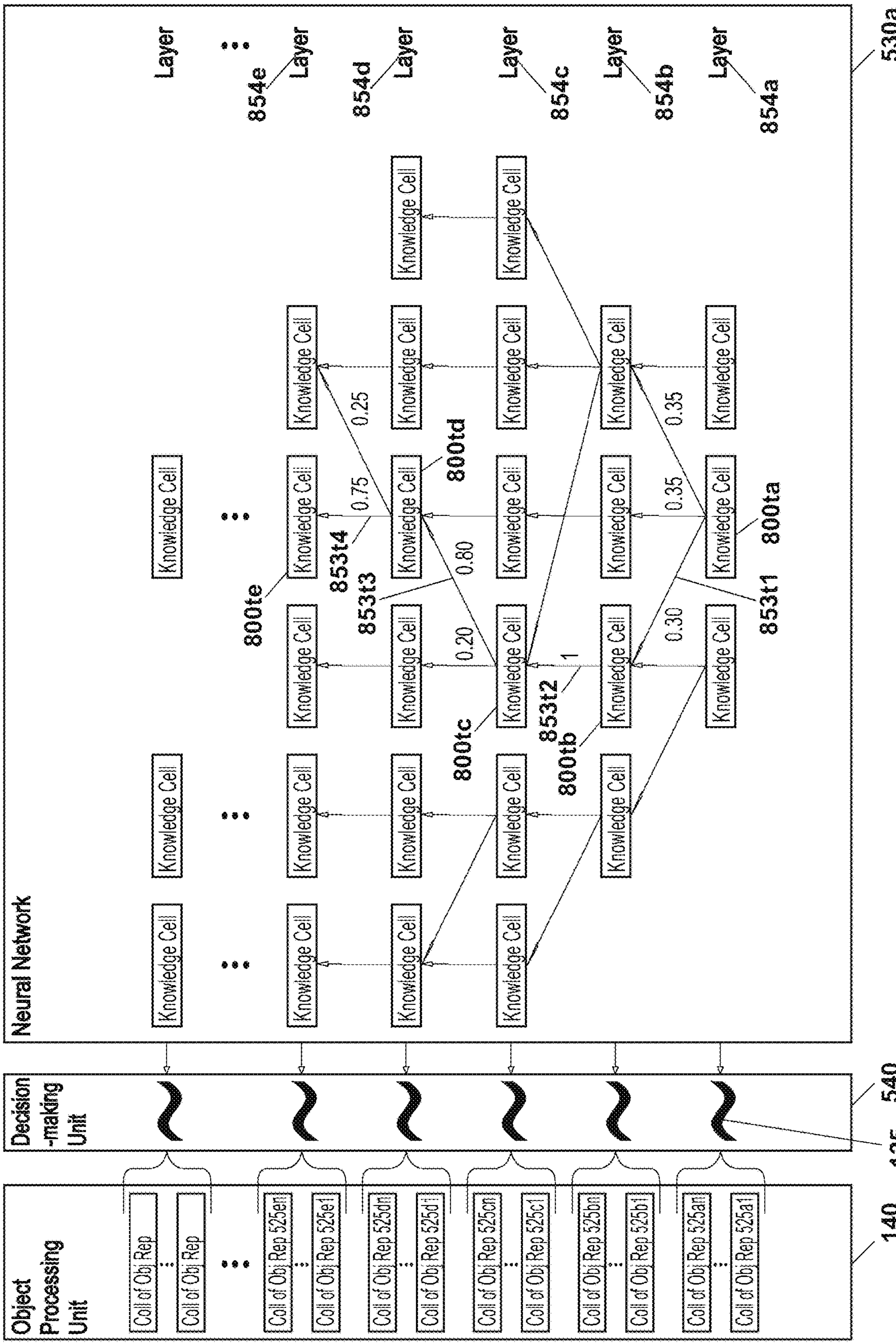

Object Processing Unit
Coll of Obj Rep
525l5
525l4
525l3
525l2
525l1
140
Decision -making Unit
540
125
Collection of Knowledge Cells
Knowledge Cell
800ra
800rb
800rc
800rd
525c1
525c2
525d1
525d2
525d3
525d4
525d5
530d

FIG. 25



Neural Network
Layer
854e
854d
854c
854b
854a
Knowledge Cell
800te
800td
800tc
800tb
800ta
853t1
853t2
853t3
853t4
0.25
0.75
0.80
0.20
0.35
0.35
0.30
1
530a
Decision-making Unit
540
125
Object Processing Unit
Coll of Obj Rep
Coll of Obj Rep 525em
Coll of Obj Rep 525e1
Coll of Obj Rep 525dm
Coll of Obj Rep 525d1
Coll of Obj Rep 525cm
Coll of Obj Rep 525c1
Coll of Obj Rep 525bm
Coll of Obj Rep 525b1
Coll of Obj Rep 525am
Coll of Obj Rep 525a1
140

FIG. 26

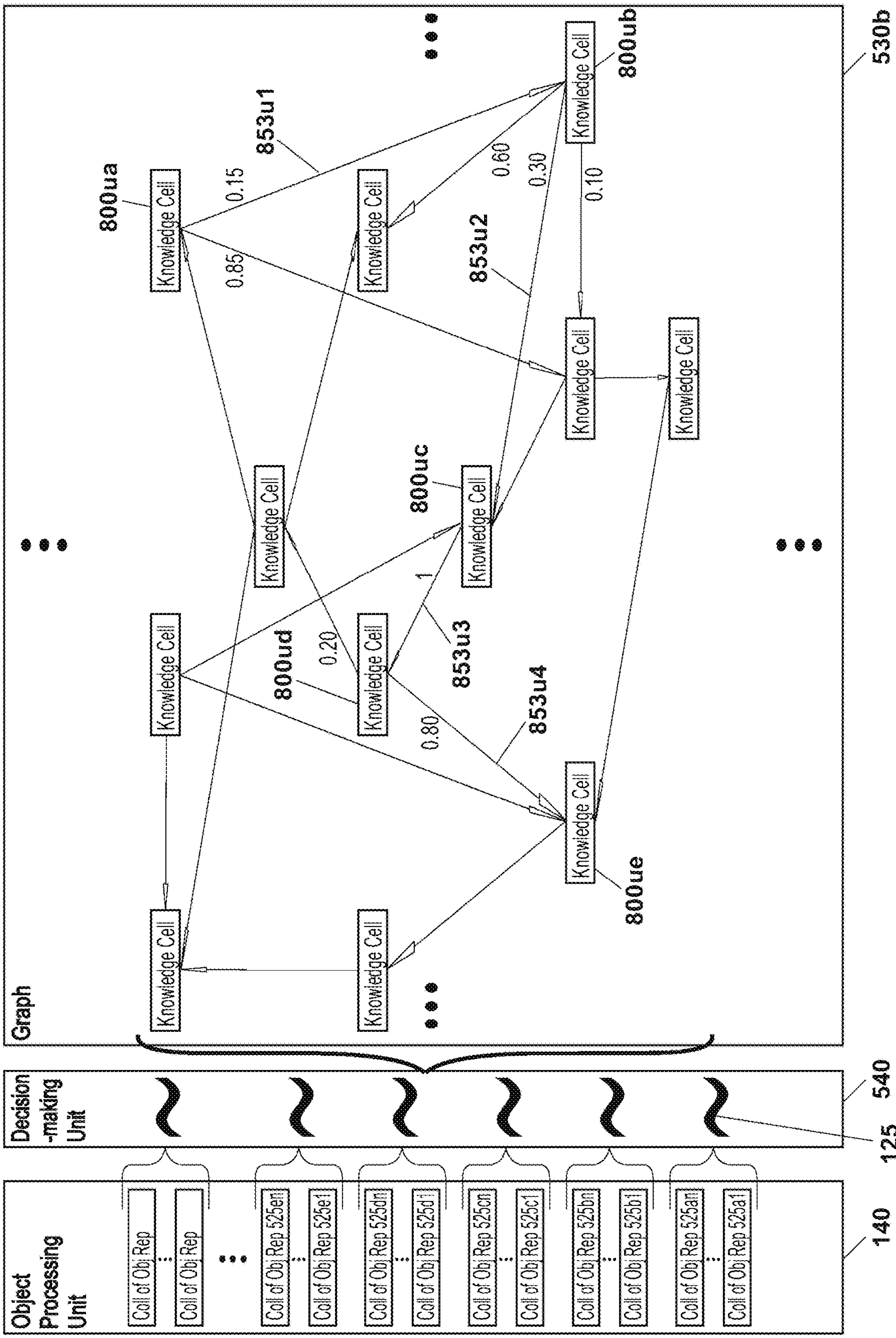

Graph
Knowledge Cell
800ua
800ub
800uc
800ud
800ue
853u1
853u2
853u3
853u4
0.15
0.85
0.60
0.30
0.10
1
0.20
0.80
530b
Decision -making Unit
540
125
Object Processing Unit
140
Coll of Obj Rep
Coll of Obj Rep 525en
Coll of Obj Rep 525e1
Coll of Obj Rep 525dn
Coll of Obj Rep 525d1
Coll of Obj Rep 525cn
Coll of Obj Rep 525c1
Coll of Obj Rep 525bn
Coll of Obj Rep 525b1
Coll of Obj Rep 525an
Coll of Obj Rep 525a1

FIG. 27

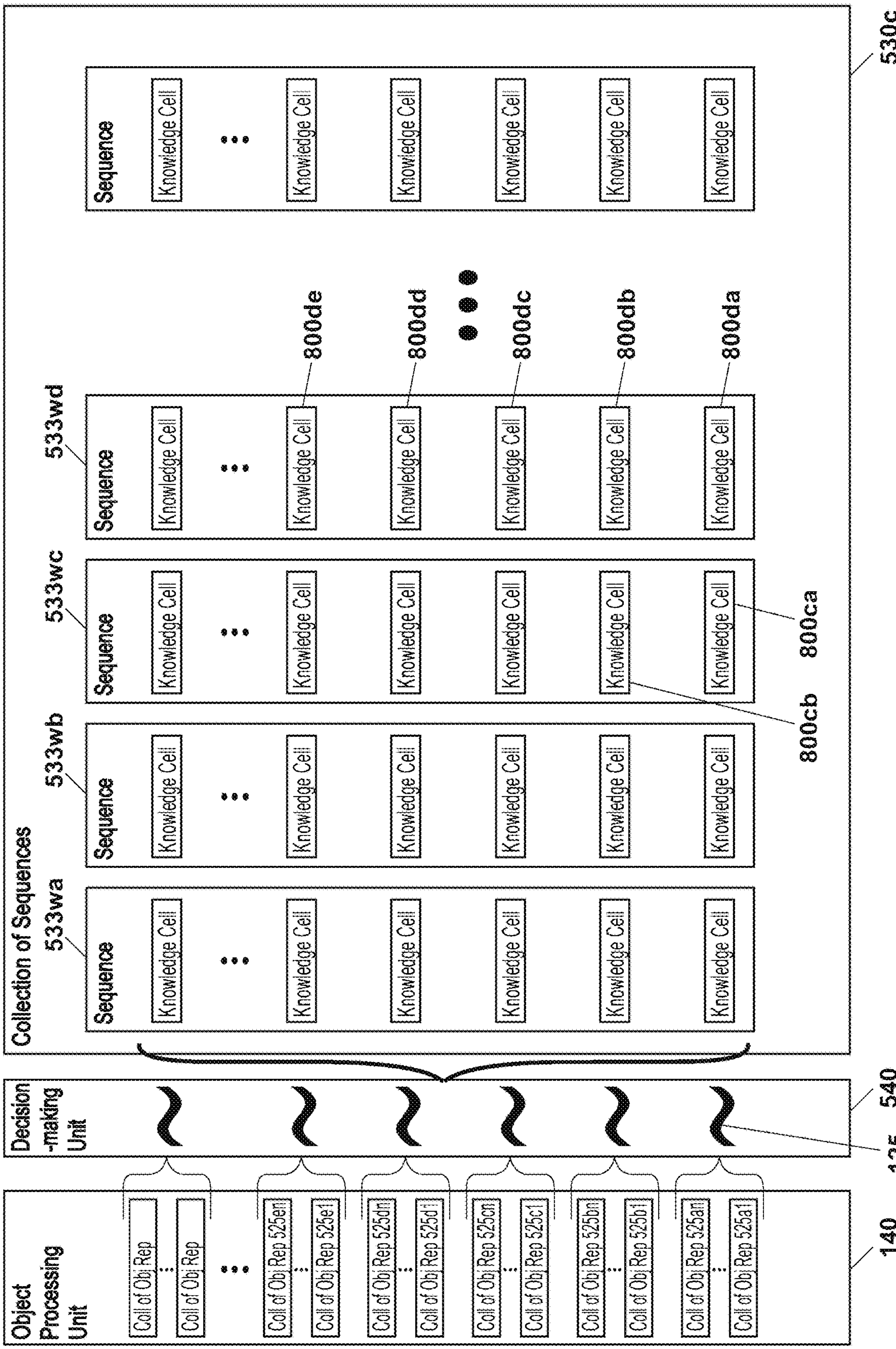

Object Processing Unit
Coll of Obj Rep
Coll of Obj Rep 525en
Coll of Obj Rep 525e1
Coll of Obj Rep 525dn
Coll of Obj Rep 525d1
Coll of Obj Rep 525cn
Coll of Obj Rep 525c1
Coll of Obj Rep 525bn
Coll of Obj Rep 525b1
Coll of Obj Rep 525an
Coll of Obj Rep 525a1
140
Decision -making Unit
540
125
Collection of Sequences
533wa
533wb
533wc
533wd
Sequence
Knowledge Cell
800ca
800cb
800da
800db
800dc
800dd
800de
530c

FIG. 28

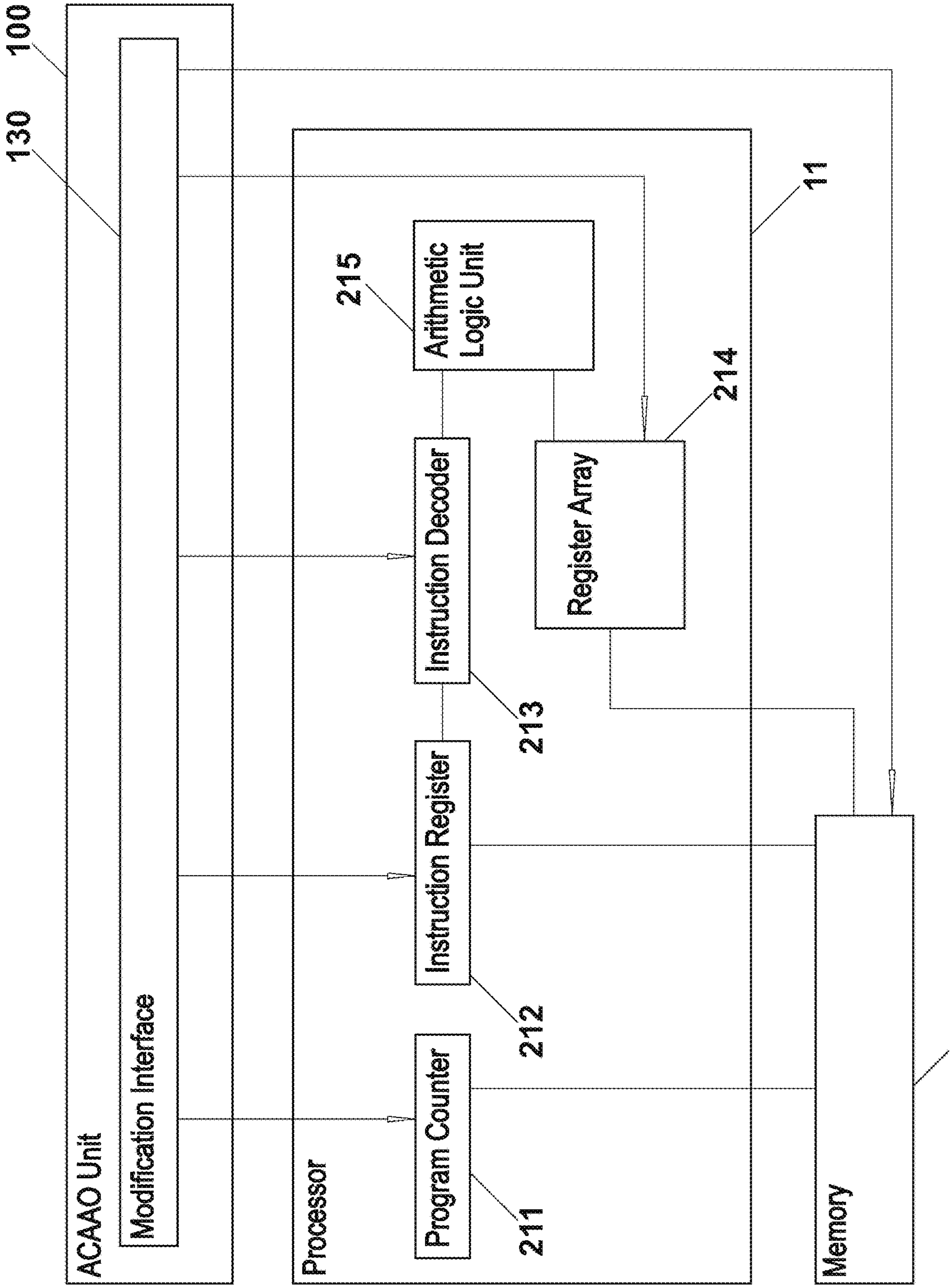

100
130
ACAAO Unit
Modification Interface
Processor
Program Counter
Instruction Register
Instruction Decoder
Arithmetic Logic Unit
Register Array
Memory
11
12
211
212
213
214
215

FIG. 29

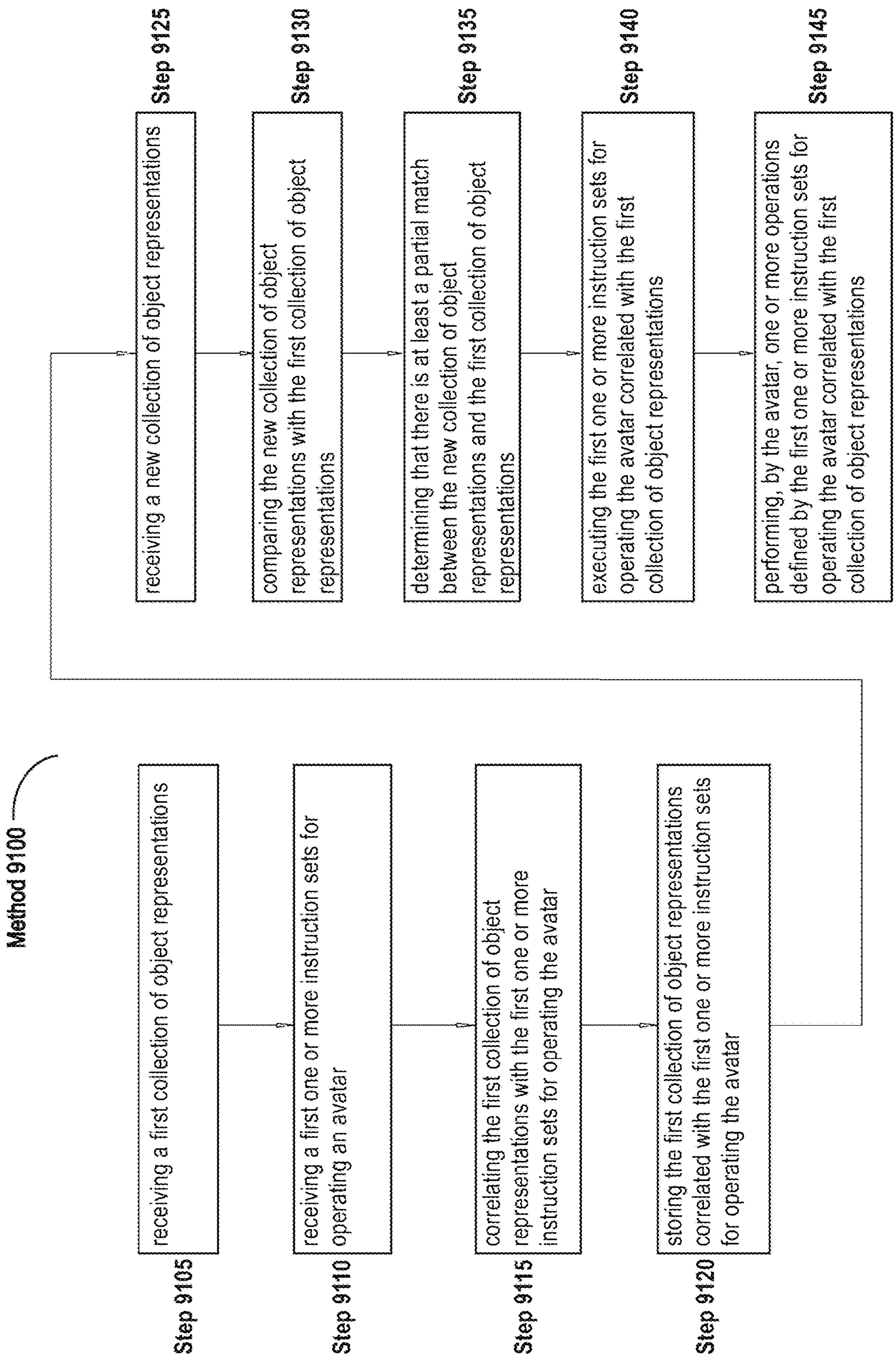

Method 9100
receiving a first collection of object representations
Step 9105
receiving a first one or more instruction sets for operating an avatar
Step 9110
correlating the first collection of object representations with the first one or more instruction sets for operating the avatar
Step 9115
storing the first collection of object representations correlated with the first one or more instruction sets for operating the avatar
Step 9120
receiving a new collection of object representations
Step 9125
comparing the new collection of object representations with the first collection of object representations
Step 9130
determining that there is at least a partial match between the new collection of object representations and the first collection of object representations
Step 9135
executing the first one or more instruction sets for operating the avatar correlated with the first collection of object representations
Step 9140
performing, by the avatar, one or more operations defined by the first one or more instruction sets for operating the avatar correlated with the first collection of object representations
Step 9145

FIG. 30

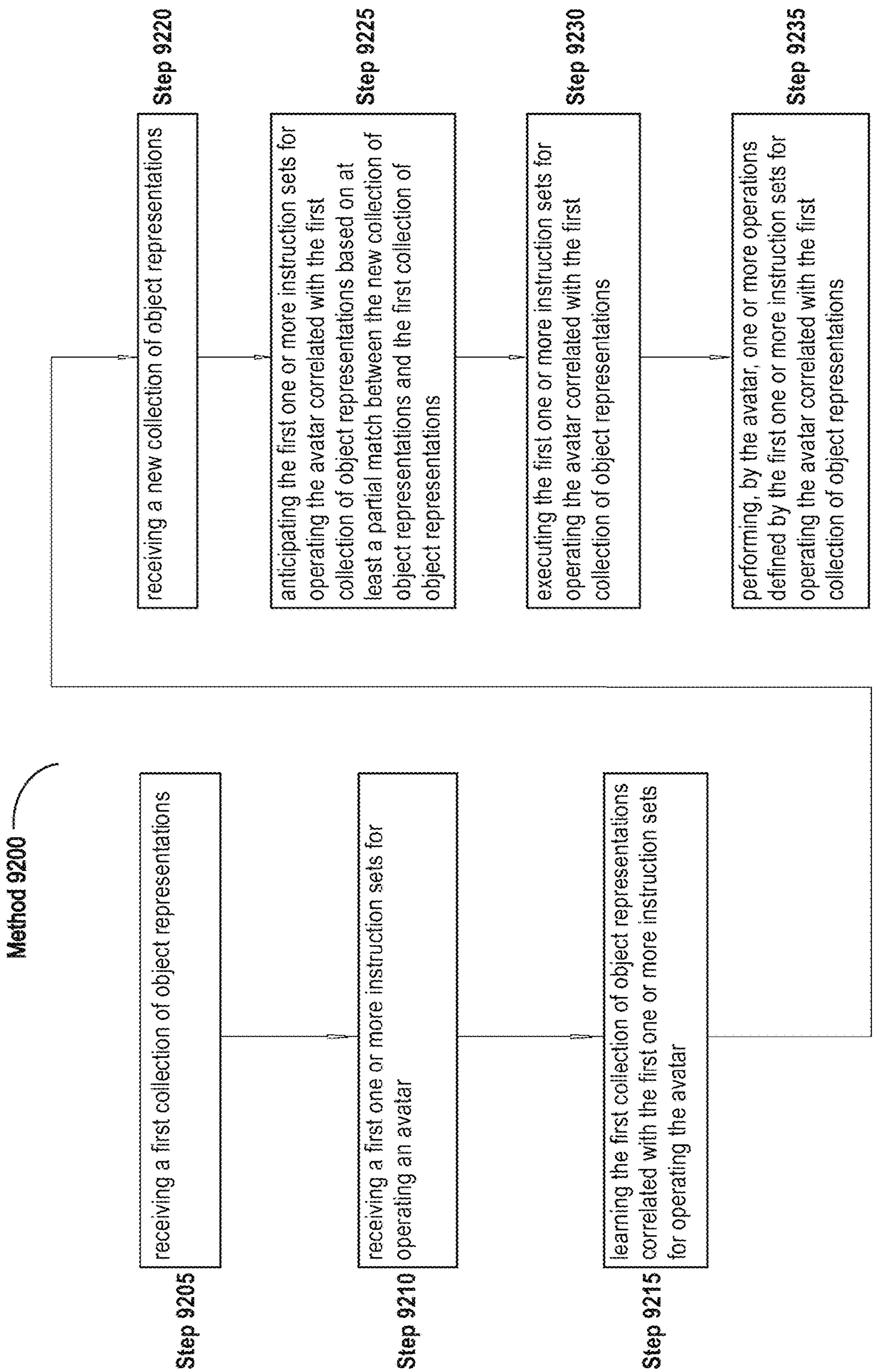

Method 9200
Step 9205
receiving a first collection of object representations
Step 9210
receiving a first one or more instruction sets for operating an avatar
Step 9215
learning the first collection of object representations correlated with the first one or more instruction sets for operating the avatar
Step 9220
receiving a new collection of object representations
Step 9225
anticipating the first one or more instruction sets for operating the avatar correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations
Step 9230
executing the first one or more instruction sets for operating the avatar correlated with the first collection of object representations
Step 9235
performing, by the avatar, one or more operations defined by the first one or more instruction sets for operating the avatar correlated with the first collection of object representations

FIG. 31

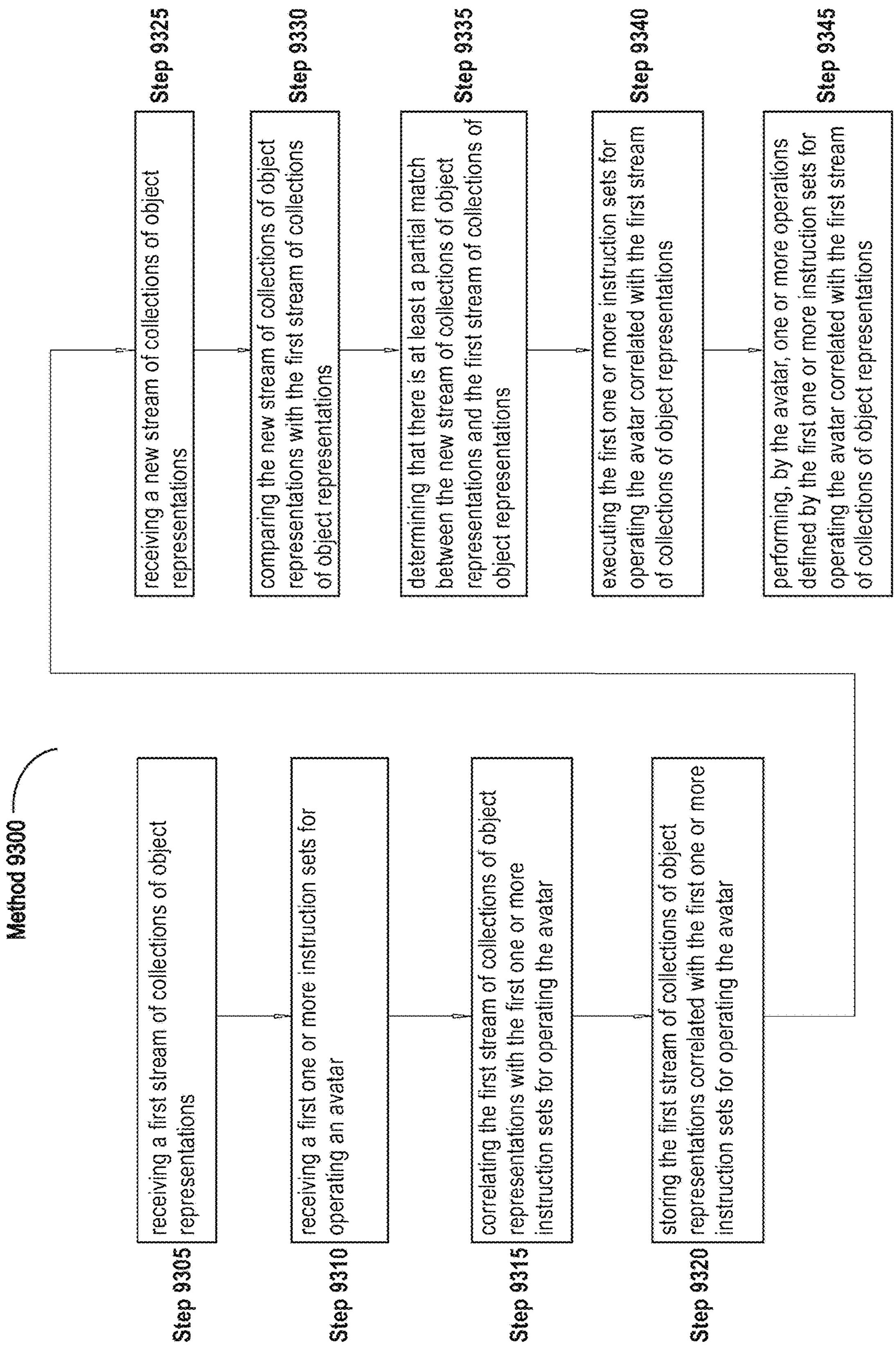

Method 9300
Step 9305
receiving a first stream of collections of object representations
Step 9310
receiving a first one or more instruction sets for operating an avatar
Step 9315
correlating the first stream of collections of object representations with the first one or more instruction sets for operating the avatar
Step 9320
storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar
Step 9325
receiving a new stream of collections of object representations
Step 9330
comparing the new stream of collections of object representations with the first stream of collections of object representations
Step 9335
determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations
Step 9340
executing the first one or more instruction sets for operating the avatar correlated with the first stream of collections of object representations
Step 9345
performing, by the avatar, one or more operations defined by the first one or more instruction sets for operating the avatar correlated with the first stream of collections of object representations

FIG. 32

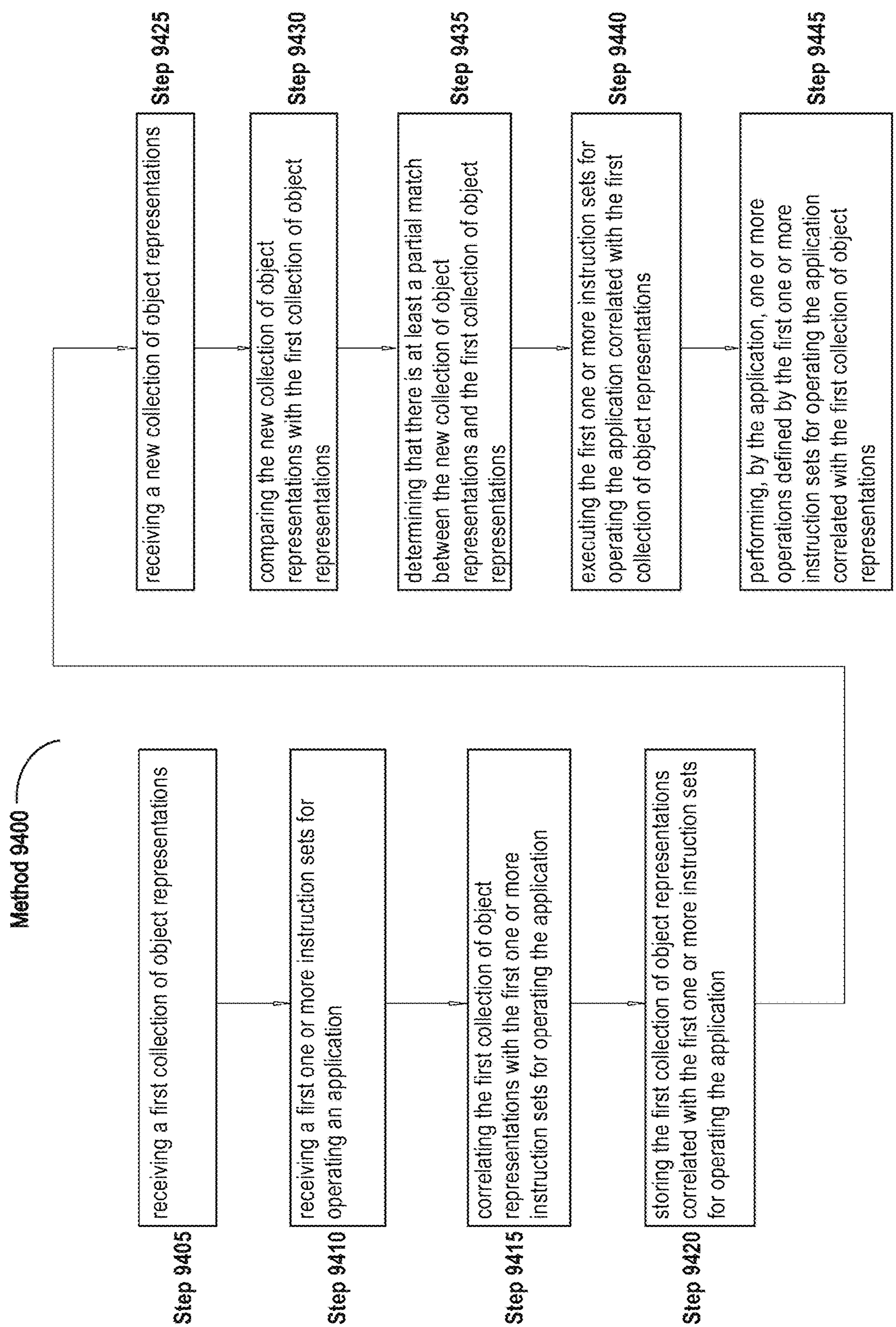

Method 9400
Step 9405
receiving a first collection of object representations
Step 9410
receiving a first one or more instruction sets for operating an application
Step 9415
correlating the first collection of object representations with the first one or more instruction sets for operating the application
Step 9420
storing the first collection of object representations correlated with the first one or more instruction sets for operating the application
Step 9425
receiving a new collection of object representations
Step 9430
comparing the new collection of object representations with the first collection of object representations
Step 9435
determining that there is at least a partial match between the new collection of object representations and the first collection of object representations
Step 9440
executing the first one or more instruction sets for operating the application correlated with the first collection of object representations
Step 9445
performing, by the application, one or more operations defined by the first one or more instruction sets for operating the application correlated with the first collection of object representations

FIG. 33

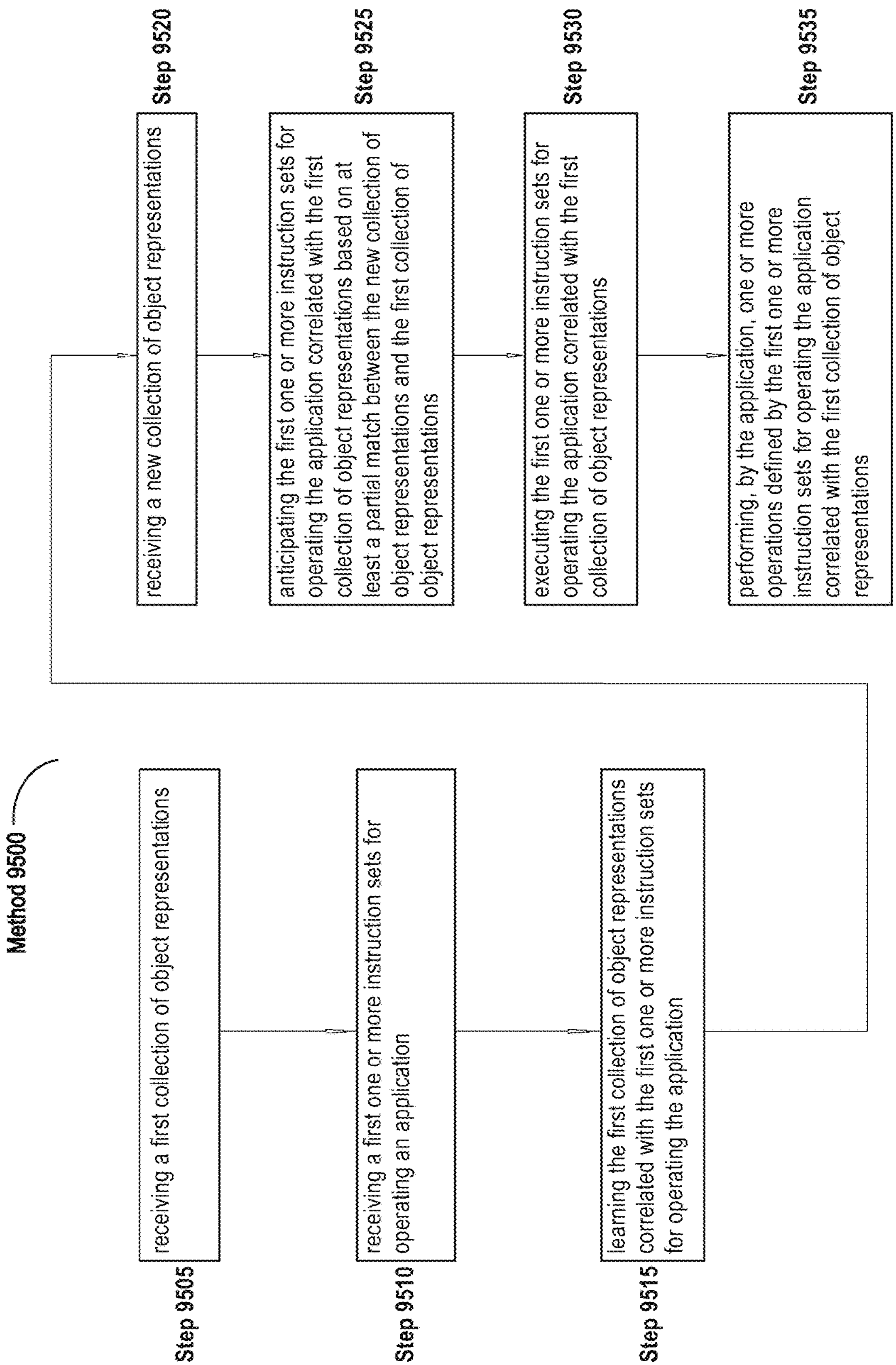

Method 9500
Step 9505
receiving a first collection of object representations
Step 9510
receiving a first one or more instruction sets for operating an application
Step 9515
learning the first collection of object representations correlated with the first one or more instruction sets for operating the application
Step 9520
receiving a new collection of object representations
Step 9525
anticipating the first one or more instruction sets for operating the application correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations
Step 9530
executing the first one or more instruction sets for operating the application correlated with the first collection of object representations
Step 9535
performing, by the application, one or more operations defined by the first one or more instruction sets for operating the application correlated with the first collection of object representations

FIG. 34

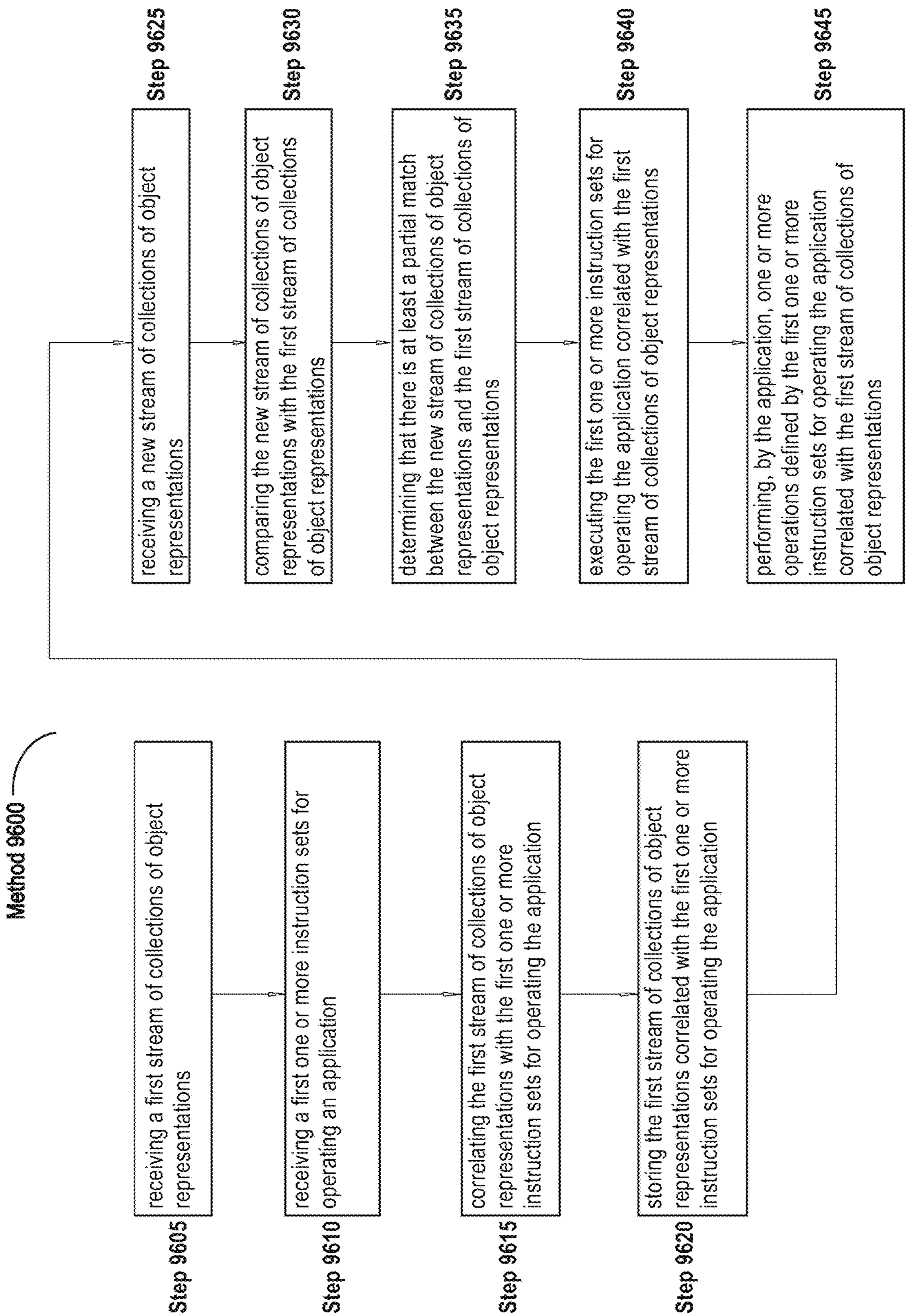

Method 9600
Step 9605
receiving a first stream of collections of object representations
Step 9610
receiving a first one or more instruction sets for operating an application
Step 9615
correlating the first stream of collections of object representations with the first one or more instruction sets for operating the application
Step 9620
storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the application
Step 9625
receiving a new stream of collections of object representations
Step 9630
comparing the new stream of collections of object representations with the first stream of collections of object representations
Step 9635
determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations
Step 9640
executing the first one or more instruction sets for operating the application correlated with the first stream of collections of object representations
Step 9645
performing, by the application, one or more operations defined by the first one or more instruction sets for operating the application correlated with the first stream of collections of object representations

FIG. 35

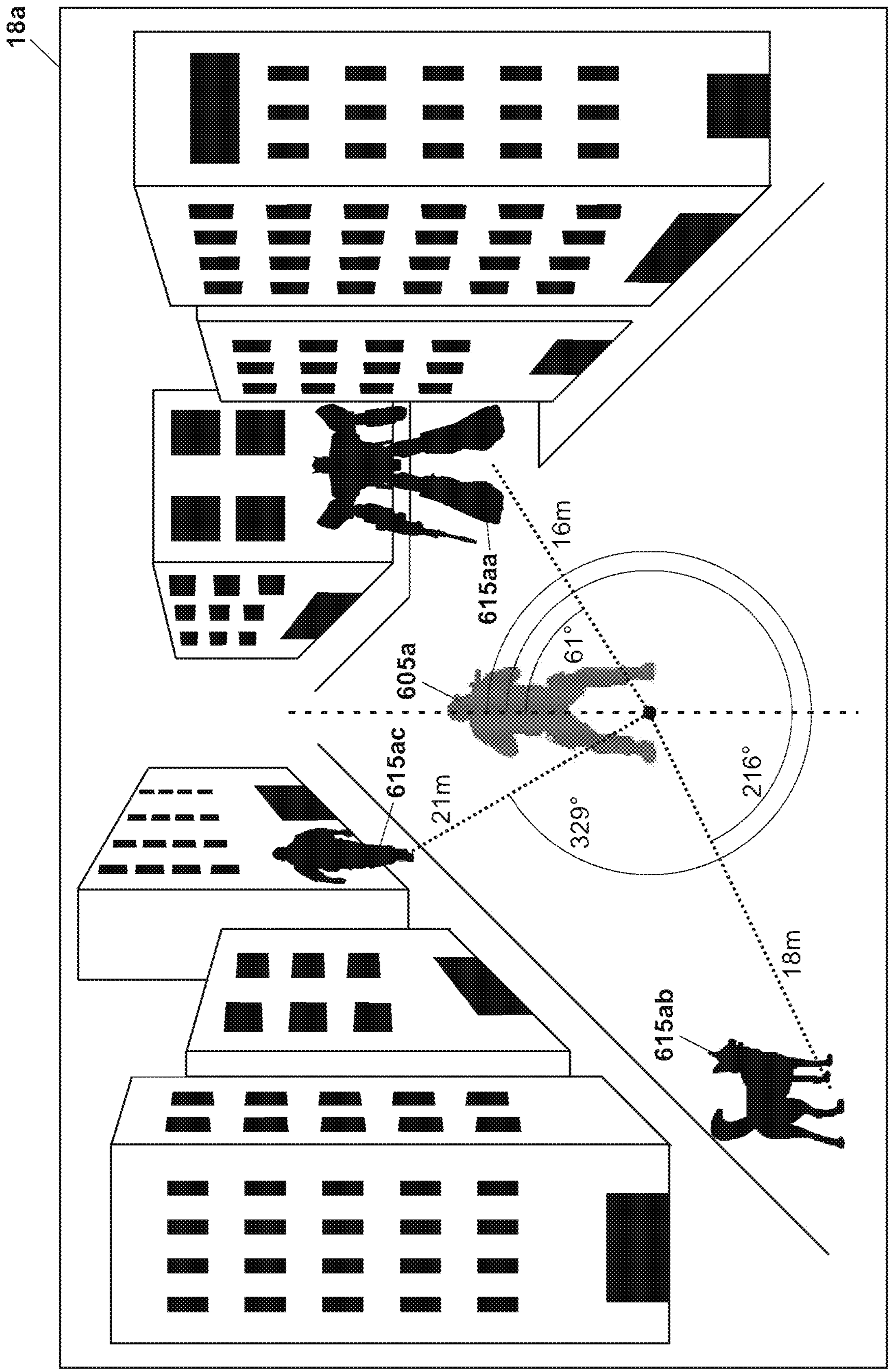


FIG. 36

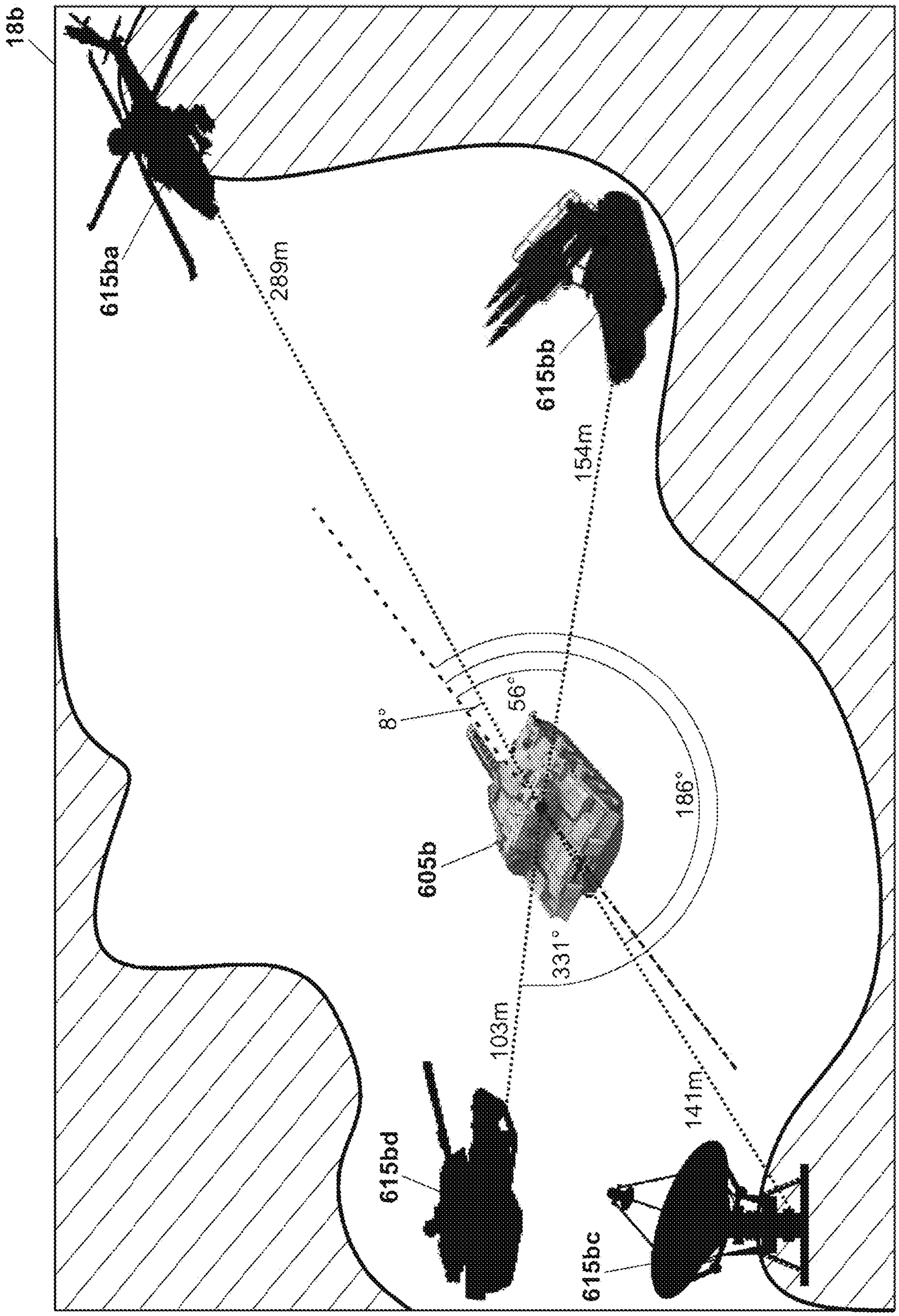

18b
615ba
289m
615bb
154m
8°
56°
186°
605b
331°
103m
141m
615bd
615bc

FIG. 37

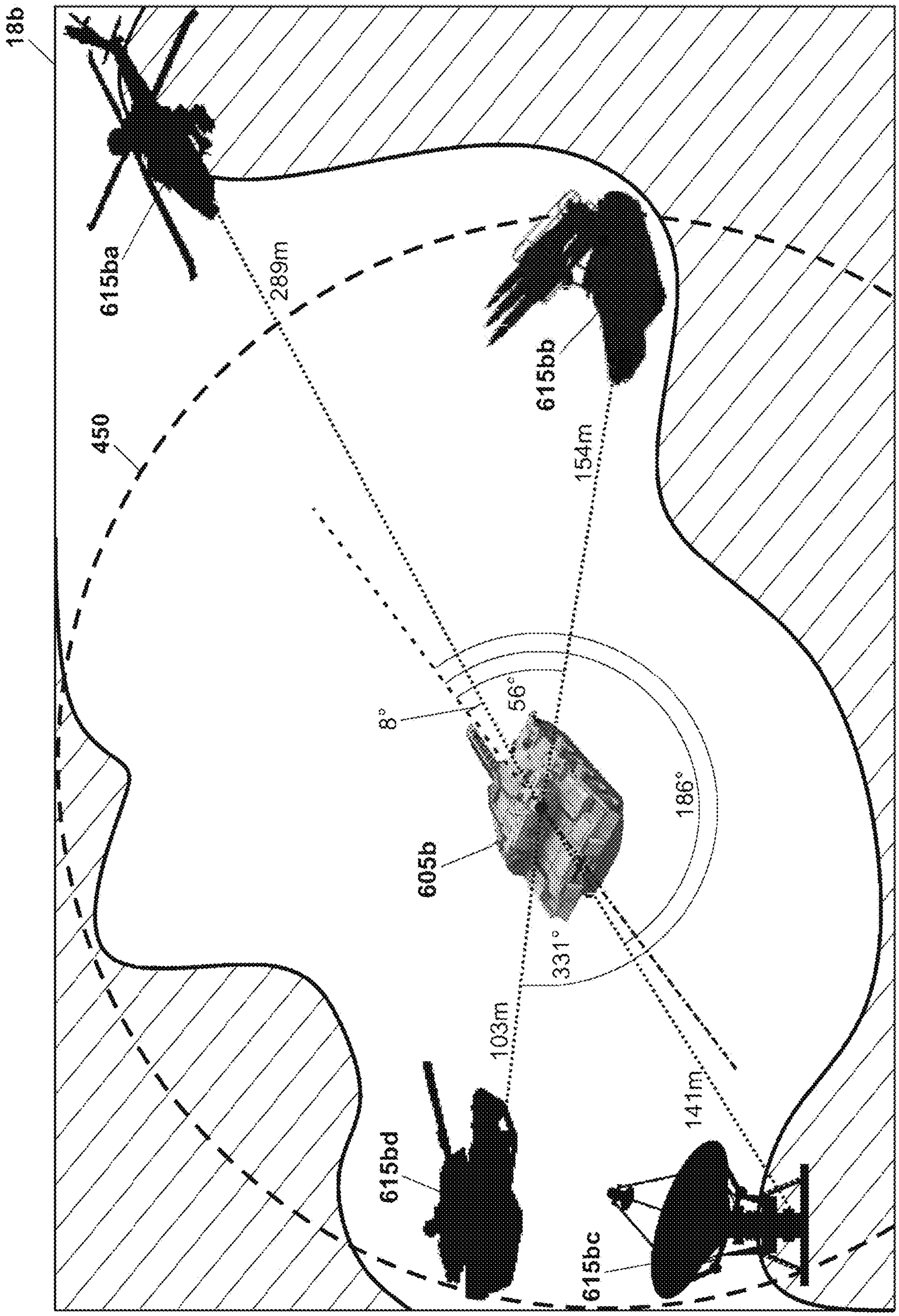

18b
615ba
289m
450
615bb
154m
8°
56°
186°
605b
331°
103m
141m
615bd
615bc

FIG. 38

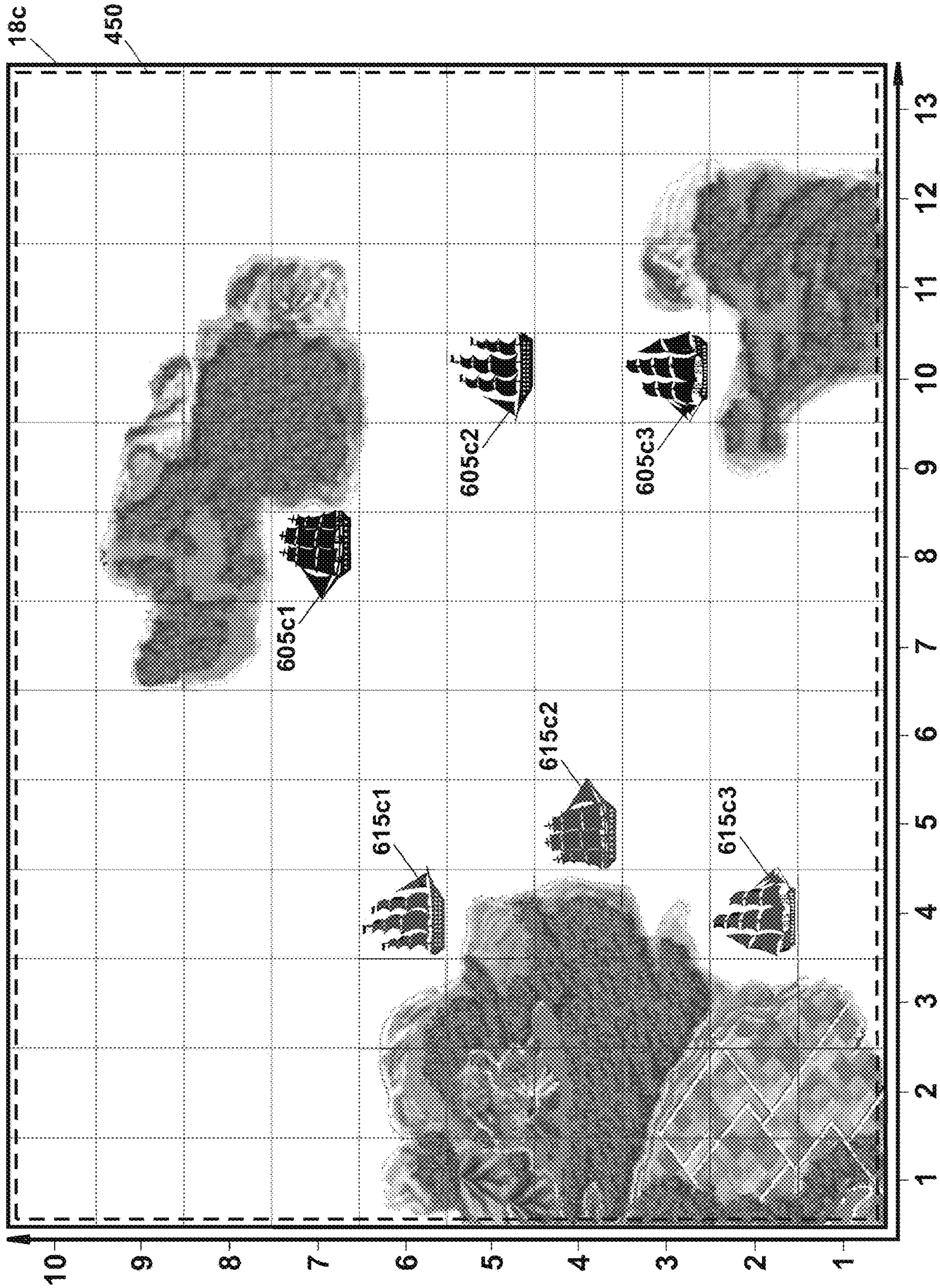

18c
450
605c1
605c2
605c3
615c1
615c2
615c3
1
2
3
4
5
6
7
8
9
10
11
12
13
1
2
3
4
5
6
7
8
9
10

FIG. 39

ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING AN AVATAR'S CIRCUMSTANCES FOR AUTONOMOUS AVATAR OPERATION

FIELD

The disclosure generally relates to computing devices and/or systems. The disclosure includes devices, apparatuses, systems, and related methods for providing advanced learning, anticipating, decision making, automation, and/or other functionalities.

COPYRIGHT NOTICE

BACKGROUND

Applications and/or avatars thereof commonly operate by receiving a user's operating directions in various circumstances. Instructions are then executed to effect the operation of an application and/or avatar based on user's operating directions. Hence, applications and/or avatars rely on the user to direct their behaviors. Commonly employed application and/or avatar operating techniques lack a way to learn operation of an application and/or avatar and enable autonomous operation of an application and/or avatar.

SUMMARY

In some aspects, the disclosure relates to a system for learning and using an avatar's circumstances for autonomous avatar operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets of an application. The system may further comprise: a memory unit configured to store data. The system may further comprise: an artificial intelligence unit configured to. The artificial intelligence unit may be configured to: receive a first collection of object representations, the first collection of object representations including one or more object representations representing one or more objects of the application. The artificial intelligence unit may be further **30** configured to: receive a first one or more instruction sets for operating an avatar of the application. The artificial intelligence unit may be further configured to: learn the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application. The artificial intelligence unit may be further configured to: receive a new collection of object representations, the new collection of object representations including one or more object representations representing one or more objects of the application. The artificial intelligence unit may be further configured to: anticipate the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations.

The artificial intelligence unit may be further configured to: cause the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the avatar of the application performs one or more operations defined by the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations.

In certain embodiments, the processor circuit includes one or more processor circuits. In further embodiments, the application includes a computer game, a virtual world, a 3D graphics application, a 2D graphics application, a web browser, a media application, a word processing application, a spreadsheet application, a database application, a forms-based application, an operating system, a device control application, a system control application, or a computer application. In further embodiments, at least one of: the processor circuit, the memory unit, or the artificial intelligence unit of the system are part of a single computing device.

In some embodiments, the memory unit includes one or more memory units. In further embodiments, the memory unit resides on a remote computing device or a remote computing system. The remote computing device or the remote computing system may include a server, a cloud, a computing device, or a computing system accessible over a network or an interface.

In certain embodiments, the artificial intelligence unit includes a circuit, a computing apparatus, a computing system, or a hardware element. In further embodiments, the artificial intelligence unit includes an application. In further embodiments, the artificial intelligence unit is coupled to the memory unit. In further embodiments, the artificial intelligence unit is a hardware element that is part of, an application operating on, or an element coupled to the processor circuit. In further embodiments, the artificial intelligence unit is part of or coupled to the application. In further embodiments, the artificial intelligence unit is part of or coupled to the avatar of the application. In further embodiments, the system further comprises: an additional processor circuit, wherein the artificial intelligence unit is a hardware element that is part of, an application operating on, or an element coupled to the additional processor circuit. In further embodiments, the artificial intelligence unit is a hardware element that is part of, an application operating on, or an element coupled to a remote computing device or a remote computing system. In further embodiments, the artificial intelligence unit is attachable to the processor circuit. In further embodiments, the artificial intelligence unit is attachable to the application. In further embodiments, the artificial intelligence unit is attachable to the avatar of the application. In further embodiments, the artificial intelligence unit is embedded or built into the processor circuit. In further embodiments, the artificial intelligence unit is embedded or built into the application. In further embodiments, the artificial intelligence unit is embedded or built into the avatar of the application. In further embodiments, the artificial intelligence unit is provided as a feature of the processor circuit. In further embodiments, the artificial intelligence unit is provided as a feature of the application. In further embodiments, the artificial intelligence unit is provided as a feature of the avatar of the application. In further embodiments, the artificial intelligence unit is further configured to: take control from, share control with, or release control to the processor circuit. In further embodiments, the

artificial intelligence unit is further configured to: take control from, share control with, or release control to the application. In further embodiments, the artificial intelligence unit is further configured to: take control from, share control with, or release control to the avatar of the application.

In some embodiments, the one or more objects of the application include a 2D model, a 3D model, a 2D shape, a 3D shape, a graphical user interface element, a form element, a data or database element, a spreadsheet element, a link, a picture, a text, a number, or a computer object. In further embodiments, the one or more objects of the application include one or more objects of the application in the avatar's surrounding. The avatar's surrounding may include an area of interest around the avatar. In further embodiments, the avatar of the application includes a user-controllable object of the application. In further embodiments, an avatar's circumstance includes one or more objects of the application.

In certain embodiments, the first collection of object representations is received at a first time. In further embodiments, the new collection of object representations is received at a new time. In further embodiments, the first collection of object representations includes a unit of knowledge of the avatar's circumstance at a first time. In further embodiments, the new collection of object representations includes a unit of knowledge of the avatar's circumstance at a new time. In further embodiments, an object representation includes one or more properties of an object of the application. In further embodiments, an object representation includes one or more information on an object of the application. In further embodiments, the first or the new collection of object representations includes or is associated with a time stamp, an order, or a time related information. In further embodiments, the first collection of object representations includes a comparative collection of object representations whose at least one portion can be used for comparisons with at least one portion of collections of object representations subsequent to the first collection of object representations, the collections of object representations subsequent to the first collection of object representations comprising the new collection of object representations. In further embodiments, the first collection of object representations includes a comparative collection of object representations that can be used for comparison with the new collection of object representations. In further embodiments, the new collection of object representations includes an anticipatory collection of object representations that can be compared with collections of object representations whose correlated one or more instruction sets for operating the avatar of the application can be used for anticipation of one or more instruction sets to be executed in autonomous operating of the avatar of the application. In further embodiments, the first collection of object representations includes a stream of collections of object representations. In further embodiments, the new collection of object representations includes a stream of collections of object representations.

In some embodiments, the receiving the first collection of object representations includes receiving one or more properties of the one or more objects of the application. The one or more properties of the one or more objects of the application may include one or more information on the one or more objects of the application. The receiving the one or more properties of the one or more objects of the application may include receiving the one or more properties of the one or more objects of the application from an engine, an environment, or a system used to implement the application.

The receiving the one or more properties of the one or more objects of the application may include at least one of: accessing or reading a scene graph or a data structure used for organizing the one or more objects of the application. The receiving the one or more properties of the one or more objects of the application may include detecting the one or more properties of the one or more objects of the application in a picture of the avatar's surrounding. The receiving the one or more properties of the one or more objects of the application may include detecting the one or more properties of the one or more objects of the application in a sound from the avatar's surrounding.

In certain embodiments, the system further comprises: an object processing unit configured to receive collections of object representations, wherein the first or the new collection of object representations is received by the object processing unit.

In some embodiments, the first one or more instruction sets for operating the avatar of the application include one or more instruction sets that temporally correspond to the first collection of object representations. The one or more instruction sets that temporally correspond to the first collection of object representations may include one or more instruction sets executed at a time of generating the first collection of object representations. The one or more instruction sets that temporally correspond to the first collection of object representations may include one or more instruction sets executed prior to generating the first collection of object representations. The one or more instruction sets that temporally correspond to the first collection of object representations may include one or more instruction sets executed within a threshold period of time prior to generating the first collection of object representations. The one or more instruction sets that temporally correspond to the first collection of object representations may include one or more instruction sets executed subsequent to generating the first collection of object representations. The one or more instruction sets that temporally correspond to the first collection of object representations may include one or more instruction sets executed within a threshold period of time subsequent to generating the first collection of object representations. The one or more instruction sets that temporally correspond to the first collection of object representations may include one or more instruction sets executed within a threshold period of time prior to generating the first collection of object representations and a threshold period of time subsequent to generating the first collection of object representations.

In certain embodiments, the first one or more instruction sets for operating the avatar of the application include one or more instruction sets executed in operating the avatar of the application. In further embodiments, the first one or more instruction sets for operating the avatar of the application are part of the application. In further embodiments, the first one or more instruction sets for operating the avatar of the application are part of the avatar of the application. In further embodiments, the first one or more instruction sets for operating the avatar of the application include one or more inputs into or one or more outputs from the processor circuit. In further embodiments, the first one or more instruction sets for operating the avatar of the application include a value or a state of a register or an element of the processor circuit. In further embodiments, the first one or more instruction sets for operating the avatar of the application include at least one of: a command, a keyword, a symbol, an instruction, an operator, a variable, a value, an object, a data structure, a function, a parameter, a state, a signal, an input,

an output, a character, a digit, or a reference thereto. In further embodiments, the first one or more instruction sets for operating the avatar of the application include a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, a structured query language (SQL) code, or a machine code. In further embodiments, the first one or more instruction sets for operating the avatar of the application include one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the first one or more instruction sets for operating the avatar of the application include one or more instruction sets for operating the application.

In some embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes receiving the first one or more instruction sets for operating the avatar of the application executed by the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes receiving the first one or more instruction sets for operating the avatar of the application as they are executed by the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes obtaining the first one or more instruction sets for operating the avatar of the application from the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes receiving the first one or more instruction sets for operating the avatar of the application from a register or an element of the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes receiving the first one or more instruction sets for operating the avatar of the application from at least one of: the memory unit, a virtual machine, a runtime engine, a hard drive, a storage device, a peripheral device, a network connected device, or a user. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes receiving the first one or more instruction sets for operating the avatar of the application from a plurality of processor circuits, applications, memory units, virtual machines, runtime engines, hard drives, storage devices, peripheral devices, network connected devices, or users. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes obtaining the first one or more instruction sets for operating the avatar of the application from the application. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes obtaining the first one or more instruction sets for operating the avatar of the application from the avatar of the application. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes receiving the one or more instruction sets for operating the avatar of the application at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: tracing, profiling, or instrumentation of a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, a structured query language (SQL) code, or a machine code. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: tracing, profiling, or instrumentation of a register of the processor circuit, the memory unit, a storage, or a repository where the first one or more instruction sets for operating the avatar of the application are stored. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: tracing, profiling, or instrumentation of the processor circuit, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a processing element. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: tracing, profiling, or instrumentation of the processor circuit or tracing, profiling, or instrumentation of a component of the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of tracing, profiling, or instrumentation of the application. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: tracing, profiling, or instrumentation of the avatar of the application. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: tracing, profiling, or instrumentation at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: tracing, profiling, or instrumentation of one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: tracing, profiling, or instrumentation of a user input. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: a manual, an automatic, a dynamic, or a just in time (JIT) tracing, profiling, or instrumentation. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes utilizing at least one of: a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, a logging tool, or an independent tool for obtaining instruction sets. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes utilizing an assembly language. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes utilizing a branch or a jump. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes a branch tracing or a simulation tracing. In further embodiments, the system further comprises: an interface configured to receive instruction sets, wherein the first one or more instruction sets for operating the avatar of the application are received via the interface. The interface may include an acquisition interface.

In certain embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application include a unit of knowledge of how the avatar of the application operated in a circumstance. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application are included in a neuron, a node, a vertex, or an element of a knowledgebase. The knowledgebase may

include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application are structured into a knowledge cell. In further embodiments, the knowledge cell is included in a neuron, a node, a vertex, or an element of a knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application includes correlating the first collection of object representations with the first one or more instruction sets for operating the avatar of the application. The correlating the first collection of object representations with the first one or more instruction sets for operating the avatar of the application may include generating a knowledge cell, the knowledge cell comprising the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application. The correlating the first collection of object representations with the first one or more instruction sets for operating the avatar of the application may include structuring a unit of knowledge of how the avatar of the application operated in a circumstance. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application includes learning a user's knowledge, style, or methodology of operating the avatar of the application in a circumstance.

In some embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application includes storing the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application into the memory unit, the memory unit comprising a plurality of collections of object representations correlated with one or more instruction sets for operating the avatar of the application. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the avatar of the application include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the avatar of the application are organized into a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, one or more collection of object representations correlated with one or more instruction sets for operating the avatar of the application of the plurality of collections of object representations correlated with one or more instruction sets for operating the avatar of the application are included in one or more neurons, nodes, vertices, or elements of a knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the avatar of the application include a user's knowledge, style, or methodology of operating the avatar of the application in circumstances. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the avatar of the application are stored on a remote computing device or a remote computing system. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the avatar of the application include an artificial intelligence system for knowledge structuring, storing, or representation. The artificial intelligence system for knowledge structuring, storing, or representation may include at least one of: a deep learning system, a supervised learning system, an unsupervised learning system, a neural network, a search-based system, an optimization-based system, a logic-based system, a fuzzy logic-based system, a tree-based system, a graph-based system, a hierarchical system, a symbolic system, a sub-symbolic system, an evolutionary system, a genetic system, a multi-agent system, a deterministic system, a probabilistic system, or a statistical system.

In certain embodiments, the anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes comparing at least one portion of the new collection of object representations with at least one portion of the first collection of object representations. The at least one portion of the new collection of object representations may include at least one object representation or at least one object property of the new collection of object representations. The at least one portion of the first collection of object representations may include at least one object representation or at least one object property of the first collection of object representations. In further embodiments, the anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes comparing at least one object representation from the new collection of object representations with at least one object representation from the first collection of object representations. In further embodiments, the comparing at least one object representation from the new collection of object representations with at least one object representation from the first collection of object representations includes comparing at least one object property of the at least one object representation from the new collection of object representations with at least one object property of the at least one object representation from the first collection of object representations. In further embodiments, the anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes comparing at least one object property of at least one object representation from the new collection of object representations with at least one object property of at least one object representation from the first collection of object representations.

In some embodiments, the anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object represen-

tations based on at least a partial match between the new collection of object representations and the first collection of object representations includes determining that there is at least a partial match between the new collection of object representations and the first collection of object representations. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that there is at least a partial match between one or more portions of the new collection of object representations and one or more portions of the first collection of object representations. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that a similarity between at least one portion of the new collection of object representations and at least one portion of the first collection of object representations exceeds a similarity threshold. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining a substantial similarity between at least one portion of the new collection of object representations and at least one portion of the first collection of object representations. The substantial similarity may be achieved when a similarity between the at least one portion of the new collection of object representations and the at least one portion of the first collection of object representations exceeds a similarity threshold. The substantial similarity may be achieved when a number or a percentage of matching or partially matching portions of the new collection of object representations and portions of the first collection of object representations exceeds a threshold number or threshold percentage. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that a number or a percentage of matching or partially matching object representations from the new collection of object representations and from the first collection of object representations exceeds a threshold number or threshold percentage. The matching or partially matching object representations from the new collection of object representations and from the first collection of object representations may be determined factoring in at least one of: a type of an object representation, an importance of an object representation, a threshold for a similarity in an object representation, or a threshold for a difference in an object representation. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that a number or a percentage of matching or partially matching object properties from the new collection of object representations and from the first collection of object representations exceeds a threshold number or threshold percentage. The matching or partially matching object properties from the new collection of object representations and from the first collection of object representations may be determined factoring in at least one of: an association of an object property with an object representation, a category of an object property, an importance of an object property, a threshold for a similarity in an object property, or a threshold for a difference in an object property. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that there is at least a partial match between at least one object representation from the new collection of object representations and at least one object representation from the first collection of object representations. The determining that there is at least a partial match between at least one object representation from the new collection of object representations and at least one object representation from the first collection of object representations may include determining that there is at least a partial match between at least one object property of the at least one object representation from the new collection of object representations and at least one object property of the at least one object representation from the first collection of object representations.

In certain embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations instead of or prior to an instruction set that would have been executed next. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying one or more instruction sets of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes inserting the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations into a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes redirecting the processor circuit to the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes redirecting the processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes transmitting, to the processor circuit for execution, the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes issuing an interrupt to the processor circuit and executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations following the interrupt. In further embodiments, the causing the pro-

cessor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes causing the application to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying one or more instruction sets of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes redirecting the application to the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes redirecting the application to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes causing the avatar of the application to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying one or more instruction sets of the avatar of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying the avatar of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes redirecting the avatar of the application to the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes redirecting the avatar of the application to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying at least one of: the memory unit, a register of the processor circuit, a storage, or a repository where instruction sets are stored or used. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying at least one of: an element of the processor circuit, an element of the application, an element of the avatar of the application, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of the avatar of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes utilizing an assembly language. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of

object representations includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes adding or inserting additional code into a code of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes at least one of: modifying, removing, rewriting, or overwriting a code of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes adding or inserting additional code into a code of the avatar of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes at least one of: modifying, removing, rewriting, or overwriting a code of the avatar of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of the avatar of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes implementing a user's knowledge, style, or methodology of operating the avatar of the application in a circumstance. In further embodiments, the system further comprises: an interface configured to cause execution of instruction sets, wherein the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations by the processor circuit is caused by the interface. The interface may include a modification interface.

In some embodiments, the avatar's performing the one or more operations defined by the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes implementing a user's knowledge, style, or methodology of operating the avatar of the application in a circumstance.

In certain embodiments, the artificial intelligence unit is further configured to: receive at least one extra information. In further embodiments, the at least one extra information include one or more of: a time information, a location information, a computed information, a visual information, an acoustic information, or a contextual information. In further embodiments, the at least one extra information include one or more of: an information on the avatar of the application, an information on the avatar's circumstance, an information on an object, an information on an object representation, an information on a collection of object representations, an information on an instruction set, an information on the application, an information on the processor circuit, or an information on a user. In further embodiments, the artificial intelligence unit is further configured to: learn the first collection of object representations correlated with the at least one extra information. The learning the first collection of object representations correlated with at least one extra information may include correlating the first collection of object representations with the at least one extra information. The learning the first collection of object representations correlated with at least one extra information may include storing the first collection of object representations correlated with the at least one extra information into the memory unit. In further embodiments, the anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations based on at least a partial match between an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations. The anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations based on at least a partial match between an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations may include comparing an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations. The anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations based on at least a partial match between an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations may include determining that a similarity between an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations exceeds a similarity threshold.

In some embodiments, the system of further comprises: a user interface, wherein the artificial intelligence unit is further configured to: cause the user interface to present a user with an option to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the system of further comprises: a user interface, wherein the artificial intelligence unit is further configured to: receive, via the user interface, a user's selection to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations.

In further embodiments, the artificial intelligence unit is further configured to: rate the executed first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. The rating the executed first one or more instruction sets for operating the avatar of the application correlated with the

first collection of object representations may include causing a user interface to display the executed first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations along with one or more rating values as options to be selected by a user. The rating the executed first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations may include rating the executed first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations without a user input.

In certain embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: cause the user interface to present a user with an option to cancel the execution of the executed first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the canceling the execution of the executed first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes restoring the processor circuit, the application, or the avatar of the application to a prior state. The restoring the processor circuit, the application, or the avatar of the application to a prior state may include saving the state of the processor circuit, the application, or the avatar of the application prior to executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations.

In some embodiments, the system further comprises: an input device configured to receive a user's operating directions, the user's operating directions for instructing the processor circuit, the application, or the avatar of the application on how to operate the avatar of the application.

In certain embodiments, the autonomous avatar operating includes a partially or a fully autonomous avatar operating. The partially autonomous avatar operating may include executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations responsive to a user confirmation. The fully autonomous avatar operating may include executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations without a user confirmation.

In some embodiments, the artificial intelligence unit is further configured to: receive a second collection of object representations, the second collection of object representations including one or more object representations representing one or more objects of the application; receive a second one or more instruction sets for operating an avatar of the application; and learn the second collection of object representations correlated with the second one or more instruction sets for operating the avatar of the application. In further embodiments, the second collection of object representations is received at a second time. In further embodiments, the second collection of object representations includes a unit of knowledge of the avatar's circumstance at a second time. In further embodiments, the second collection of object representations includes a stream of collections of object representations. In further embodiments, the second collection of object representations includes or is associated with a time stamp, an order, or a time related information. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application and the learning the second collection of object representations correlated with the second one or more instruction sets for operating the avatar of the application include creating a connection between the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application and the second collection of object representations correlated with the second one or more instruction sets for operating the avatar of the application. The connection may include or be associated with at least one of: an occurrence count, a weight, a parameter, or a data. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application and the learning the second collection of object representations correlated with the second one or more instruction sets for operating the avatar of the application include updating a connection between the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application and the second collection of object representations correlated with the second one or more instruction sets for operating the avatar of the application. The updating the connection between the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application and the second collection of object representations correlated with the second one or more instruction sets for operating the avatar of the application may include updating at least one of: an occurrence count, a weight, a parameter, or a data included in or associated with the connection. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application includes storing the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application into a first node of a knowledgebase, and wherein the learning the second collection of object representations correlated with the second one or more instruction sets for operating the avatar of the application includes storing the second collection of object representations correlated with the second one or more instruction sets for operating the avatar of the application into a second node of the knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. The knowledgebase may be stored in the memory unit. The learning the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application and the learning the second collection of object representations correlated with the second one or more instruction sets for operating the avatar of the application may include creating a connection between the first node and the second node. The learning the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application and the learning the second collection of object representations correlated with the second one or more instruction sets for operating the avatar of the application may include updating a connection between the first node and the second node. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application is stored into a first node of a neural network and the second collection of object representations correlated with the second one or more instruction sets for operating the avatar of the application is stored into a second node of the neural network. The first node and the second node may be

connected by a connection. The first node may be part of a first layer of the neural network and the second node may be part of a second layer of the neural network. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application is stored into a first node of a graph and the second collection of object representations correlated with the second one or more instruction sets for operating the avatar of the application is stored into a second node of the graph. The first node and the second node may be connected by a connection. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application is stored into a first node of a sequence and the second collection of object representations correlated with the second one or more instruction sets for operating the avatar of the application is stored into a second node of the sequence.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first collection of object representations, the first collection of object representations including one or more object representations representing one or more objects of an application. The operations may further comprise: receiving a first one or more instruction sets for operating an avatar of the application. The operations may further comprise: learning the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application. The operations may further comprise: receiving a new collection of object representations, the new collection of object representations including one or more object representations representing one or more objects of the application. The operations may further comprise: anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further comprise: causing an execution of the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, wherein the avatar of the application performs one or more operations defined by the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations.

In some embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes receiving the first one or more instruction sets for operating the avatar of the application from the one or more processor circuits or from another one or more processor circuits. In further embodiments, the execution of the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations is performed by the one or more processor circuits or by another one or more processor circuits.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a processor circuit, the first collection of object representations including one or more object representations representing one or more objects of an application. The method may further comprise: (b) receiving a first one or more instruction sets for operating an avatar of the application by the processor circuit. The method may further comprise: (c) learning the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application, the learning of (c) performed by the processor circuit. The method may further comprise: (d) receiving a new collection of object representations by the processor circuit, the new collection of object representations including one or more object representations representing one or more objects of the application. The method may further comprise: (e) anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (e) performed by the processor circuit. The method may further comprise: (f) executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations, the executing of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the avatar of the application, one or more operations defined by the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations.

In certain embodiments, the receiving of (b) includes receiving the first one or more instruction sets for operating the avatar of the application from the processor circuit or from another processor circuit. In further embodiments, the executing of (f) is performed by the processor circuit or by another processor circuit.

The aforementioned system, the non-transitory computer storage medium, and/or the method may include any elements, operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable as well as the following embodiments.

In some embodiments, the first one or more instruction sets for operating the avatar of the application include one or more instruction sets executed in operating the avatar of the application. In further embodiments, the first one or more instruction sets for operating the avatar of the application are part of the application. In further embodiments, the first one or more instruction sets for operating the avatar of the application are part of the avatar of the application. In further embodiments, the first one or more instruction sets for operating the avatar of the application include one or more inputs into or one or more outputs from a processor circuit. In further embodiments, the first one or more instruction sets for operating the avatar of the application include a value or a state of a register or an element of a processor circuit. In further embodiments, the first one or more instruction sets for operating the avatar of the application include at least one of: a command, a keyword, a symbol, an instruction, an operator, a variable, a value, an object, a data structure, a function, a parameter, a state, a signal, an input, an output, a character, a digit, or a reference thereto. In further embodiments, the first one or more instruction sets for operating the avatar of the application include a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, a structured query language (SQL) code, or a machine code. In further embodiments, the first one or

more instruction sets for operating the avatar of the application include one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the first one or more instruction sets for operating the avatar of the application include one or more instruction sets for operating the application.

In certain embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes receiving the first one or more instruction sets for operating the avatar of the application executed by a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes receiving the first one or more instruction sets for operating the avatar of the application as they are executed by a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes obtaining the first one or more instruction sets for operating the avatar of the application from a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes receiving the first one or more instruction sets for operating the avatar of the application from a register or an element of a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes receiving the first one or more instruction sets for operating the avatar of the application from at least one of: the memory unit, a virtual machine, a runtime engine, a hard drive, a storage device, a peripheral device, a network connected device, or a user. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes receiving the first one or more instruction sets for operating the avatar of the application from a plurality of processor circuits, applications, memory units, virtual machines, runtime engines, hard drives, storage devices, peripheral devices, network connected devices, or users. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes obtaining the first one or more instruction sets for operating the avatar of the application from the application. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes obtaining the first one or more instruction sets for operating the avatar of the application from the avatar of the application. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes receiving the one or more instruction sets for operating the avatar of the application at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of tracing, profiling, or instrumentation of a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, a structured query language (SQL) code, or a machine code. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: tracing, profiling, or instrumentation of a register of a processor circuit, a memory unit, a storage, or a repository where the first one or more instruction sets for operating the avatar of the application are stored. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: tracing, profiling, or instrumentation of a processor circuit, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a processing element. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: tracing, profiling, or instrumentation of a processor circuit or tracing, profiling, or instrumentation of a component of a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: tracing, profiling, or instrumentation of the application. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: tracing, profiling, or instrumentation of the avatar of the application. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: tracing, profiling, or instrumentation at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: tracing, profiling, or instrumentation of one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: tracing, profiling, or instrumentation of a user input. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: a manual, an automatic, a dynamic, or a just in time (JIT) tracing, profiling, or instrumentation. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes utilizing at least one of: a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, a logging tool, or an independent tool for obtaining instruction sets. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes utilizing an assembly language. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes utilizing a branch or a jump. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes a branch tracing or a simulation tracing. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes receiving the first one or more instruction sets for operating the avatar of the application via an interface. The interface may include an acquisition interface.

In some embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application includes storing the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application into a memory unit, the memory unit comprising a plurality of collections of object representations correlated with one or more instruction sets for operating the avatar of the application.

In certain embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations instead of or prior to an instruction set that would have been executed

next. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying one or more instruction sets of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying a register or an element of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes inserting the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations into a register or an element of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes redirecting a processor circuit to the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes redirecting a processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes transmitting, to a processor circuit for execution, the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes issuing an interrupt to a processor circuit and executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations following the interrupt. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes causing the application to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying one or more instruction sets of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes redirecting the application to the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes redirecting the application to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes causing the avatar of the application to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying one or more instruction sets of the avatar of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying the avatar of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes redirecting the avatar of the application to the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes redirecting the avatar of the application to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying at least one of: a memory unit, a register of a processor circuit, a storage, or a repository where instruction sets are stored or used. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying at least one of: an element of a processor circuit, an element of the application, an element of the avatar of the application, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of the application. In further embodiments, the execut-

ing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of the avatar of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes utilizing an assembly language. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes adding or inserting additional code into a code of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes at least one of: modifying, removing, rewriting, or overwriting a code of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes adding or inserting additional code into a code of the avatar of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes at least one of: modifying, removing, rewriting, or overwriting a code of the avatar of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes at least one of:

branching, redirecting, extending, or hot swapping a code of the avatar of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes implementing a user's knowledge, style, or methodology of operating the avatar of the application in a circumstance. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations via an interface. The interface may include a modification interface.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving at least one extra information. In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: learning the first collection of object representations correlated with the at least one extra information. In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: presenting, via a user interface, a user with an option to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving, via a user interface, a user's selection to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: rating the executed first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: presenting, via a user interface, a user with an option to cancel the execution of the executed first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving, via an input device, a user's operating directions, the user's operating directions for instructing the processor circuit, the application, or the avatar of the application on how to operate the avatar of the application. In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving a second collection of object representations, the second collection of object representations including one or more object representations representing one or more objects of the application; receiving a second one or more instruction sets for operating the avatar of the application; and learning the second collection of object representations correlated with the second one or more instruction sets for operating the avatar of the application.

In some aspects, the disclosure relates to a system for learning an avatar's circumstances for autonomous avatar operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets of an application. The system may further comprise: a memory unit configured to store data. The system may further comprise: an artificial intelligence

unit. The artificial intelligence unit may be configured to: receive a first collection of object representations, the first collection of object representations including one or more object representations representing one or more objects of the application. The artificial intelligence unit may be further configured to: receive a first one or more instruction sets for operating an avatar of the application. The artificial intelligence unit may be further configured to: learn the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first collection of object representations, the first collection of object representations including one or more object representations representing one or more objects of an application. The operations may further comprise: receiving a first one or more instruction sets for operating an avatar of the application. The operations may further comprise: learning the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a processor circuit, the first collection of object representations including one or more object representations representing one or more objects of an application. The method may further comprise: (b) receiving a first one or more instruction sets for operating an avatar of the application by the processor circuit. The method may further comprise: (c) learning the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application, the learning of (c) performed by the processor circuit.

The aforementioned system, the non-transitory computer storage medium, and/or the method may include any elements, operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable as well as the following embodiments.

In some aspects, the disclosure relates to a system for using an avatar's circumstances for autonomous avatar operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets of an application. The system may further comprise: a memory unit configured to store data. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: access the memory unit that comprises a plurality of collections of object representations correlated with one or more instruction sets for operating an avatar of the application, the plurality of collections of object representations correlated with one or more instruction sets for operating the avatar of the application including a first collection of object representations correlated with a first one or more instruction sets for operating the avatar of the application. The artificial intelligence unit may be further configured to: receive a new collection of object representations, the new collection of object representations including one or more object representations representing one or more objects of the application. The artificial intelligence unit may be further configured to: anticipate the first one or more instruction sets for operating the avatar of the application correlated with the

first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the avatar of the application performs one or more operations defined by the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: accessing a memory unit that comprises a plurality of collections of object representations correlated with one or more instruction sets for operating an avatar of an application, the plurality of collections of object representations correlated with one or more instruction sets for operating the avatar of the application including a first collection of object representations correlated with a first one or more instruction sets for operating the avatar of the application. The operations may further comprise: receiving a new collection of object representations, the new collection of object representations including one or more object representations representing one or more objects of the application. The operations may further comprise: anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further comprise: causing an execution of the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, wherein the avatar of the application performs one or more operations defined by the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations.

In some aspects, the disclosure relates to a method comprising: (a) accessing a memory unit that comprises a plurality of collections of object representations correlated with one or more instruction sets for operating an avatar of an application, the plurality of collections of object representations correlated with one or more instruction sets for operating the avatar of the application including a first collection of object representations correlated with a first one or more instruction sets for operating the avatar of the application, the accessing of (a) performed by a processor circuit. The method may further comprise: (b) receiving a new collection of object representations by the processor circuit, the new collection of object representations including one or more object representations representing one or more objects of the application. The method may further comprise: (c) anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations based on at

least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (c) performed by the processor circuit. The method may further comprise: (d) executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations, the executing of (d) performed in response to the anticipating of (c). The method may further comprise: (e) performing, by the avatar of the application, one or more operations defined by the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations.

The aforementioned system, the non-transitory computer storage medium, and/or the method may include any elements, operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable as well as the following embodiments.

In some aspects, the disclosure relates to a system for learning and using an avatar's circumstances for autonomous avatar operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets of an application. The system may further comprise: a memory unit configured to store data. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first stream of collections of object representations, the first stream of collections of object representations including one or more object representations representing one or more objects of the application. The artificial intelligence unit may be further configured to: receive a first one or more instruction sets for operating an avatar of the application. The artificial intelligence unit may be further configured to: learn the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application. The artificial intelligence unit may be further configured to: receive a new stream of collections of object representations, the new stream of collections of object representations including one or more object representations representing one or more objects of the application. The artificial intelligence unit may be further configured to: anticipate the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the avatar of the application performs one or more operations defined by the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations.

In certain embodiments, the first stream of collections of object representations includes one or more collections of object representations, and wherein each collection of object representations of the first stream of collections of object representations includes one or more object representations. In further embodiments, the new stream of collections of object representations includes one or more collections of object representations, and wherein each collection of object representations of the new stream of collections of object representations includes one or more object representations. In further embodiments, the first stream of collections of object representations is received over a first time period. In further embodiments, the new stream of collections of object representations is received over a new time period. In further embodiments, the first stream of collections of object representations includes a unit of knowledge of the avatar's circumstance over a first time period. In further embodiments, the new stream of collections of object representations includes a unit of knowledge of the avatar's circumstance over a new time period. In further embodiments, an object representation includes one or more properties of an object of the application. In further embodiments, an object representation includes one or more information on an object of the application. In further embodiments, the first or the new stream of collections of object representations includes or is associated with a time stamp, an order, or a time related information. In further embodiments, the first stream of collections of object representations includes a comparative stream of collections of object representations whose at least one portion can be used for comparisons with at least one portion of streams of collections of object representations subsequent to the first stream of collections of object representations, the streams of collections of object representations subsequent to the first stream of collections of object representations comprising the new stream of collections of object representations. In further embodiments, the first stream of collections of object representations includes a comparative stream of collections of object representations that can be used for comparison with the new stream of collections of object representations. In further embodiments, the new stream of collections of object representations includes an anticipatory stream of collections of object representations that can be compared with streams of collections of object representations whose correlated one or more instruction sets for operating the avatar of the application can be used for anticipation of one or more instruction sets to be executed in autonomous operating of the avatar of the application.

In some embodiments, the receiving the first stream of collections of object representations includes receiving one or more properties of the one or more objects of the application. The one or more properties of the one or more objects of the application may include one or more information on the one or more objects of the application. The receiving the one or more properties of the one or more objects of the application may include receiving the one or more properties of the one or more objects of the application from an engine, an environment, or a system used to implement the application. The receiving the one or more properties of the one or more objects of the application may include at least one of: accessing or reading a scene graph or a data structure used for organizing the one or more objects of the application. The receiving the one or more properties of the one or more objects of the application may include detecting the one or more properties of the one or more objects of the application in a picture of the avatar's surrounding. The receiving the one or more properties of the one or more objects of the application may include detecting the one or more properties of the one or more objects of the application in a sound from the avatar's surrounding.

In certain embodiments, the system further comprises: an object processing unit configured to receive streams of collections of object representations, wherein the first or the new stream of collections of object representations is received by the object processing unit.

In some embodiments, the first one or more instruction sets for operating the avatar of the application include one or more instruction sets that temporally correspond to the first stream of collections of object representations. The one or more instruction sets that temporally correspond to the first stream of collections of object representations may include one or more instruction sets executed at a time of generating the first stream of collections of object representations. The one or more instruction sets that temporally correspond to the first stream of collections of object representations may include one or more instruction sets executed prior to generating the first stream of collections of object representations. The one or more instruction sets that temporally correspond to the first stream of collections of object representations may include one or more instruction sets executed within a threshold period of time prior to generating the first stream of collections of object representations. The one or more instruction sets that temporally correspond to the first stream of collections of object representations may include one or more instruction sets executed subsequent to generating the first stream of collections of object representations. The one or more instruction sets that temporally correspond to the first stream of collections of object representations may include one or more instruction sets executed within a threshold period of time subsequent to generating the first stream of collections of object representations. The one or more instruction sets that temporally correspond to the first stream of collections of object representations may include one or more instruction sets executed within a threshold period of time prior to generating the first stream of collections of object representations and a threshold period of time subsequent to generating the first stream of collections of object representations.

In some embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application include a unit of knowledge of how the avatar of the application operated in a circumstance. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application are included in a neuron, a node, a vertex, or an element of a knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. Some of the neurons, nodes, vertices, or elements are interconnected. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application are structured into a knowledge cell. In further embodiments, the knowledge cell is included in a neuron, a node, a vertex, or an element of a knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. Some of the neurons, nodes, vertices, or elements are interconnected.

In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application includes correlating the first stream of collections of object representations with the first one or more instruction sets for operating the avatar of the application. The correlating the first stream of collections of object representations with the first one or more instruction sets for operating the avatar of the application may include generating a knowledge cell, the knowledge cell comprising the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application. The correlating the first stream of collections of object representations with the first one or more instruction sets for operating the avatar of the application may include structuring a unit of knowledge of how the avatar of the application operated in a circumstance. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application includes learning a user's knowledge, style, or methodology of operating the avatar of the application in a circumstance.

In certain embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application includes storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application into the memory unit, the memory unit comprising a plurality of streams of collections of object representations correlated with one or more instruction sets for operating the avatar of the application. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the avatar of the application include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the avatar of the application are organized into a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, one or more streams of collections of object representations correlated with one or more instruction sets for operating the avatar of the application of the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the avatar of the application are included in one or more neurons, nodes, vertices, or elements of a knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. Some of the neurons, nodes, vertices, or elements are interconnected. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the avatar of the application include a user's knowledge, style, or methodology of operating the avatar of the application in circumstances. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the avatar of the application are stored on a remote computing device or a remote computing system. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the avatar of the application include an artificial intelligence system for knowledge structuring, storing, or representation. The artificial intelligence system for knowledge structuring, storing, or representation may include at least one of: a deep learning system, a supervised learning system, an unsupervised learning system, a neural network, a search-based system, an optimization-based system, a logic-based system, a fuzzy logic-based system, a tree-based system, a graph-based system, a hierarchical system, a symbolic system, a sub-symbolic system, an

evolutionary system, a genetic system, a multi-agent system, a deterministic system, a probabilistic system, or a statistical system.

In some embodiments, the anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes comparing at least one portion of the new stream of collections of object representations with at least one portion of the first stream of collections of object representations. In further embodiments, the at least one portion of the new stream of collections of object representations include at least one collection of object representations, at least one object representation, or at least one object property of the new stream of collections of object representations. In further embodiments, the at least one portion of the first stream of collections of object representations include at least one collection of object representations, at least one object representation, or at least one object property of the first stream of collections of object representations. In further embodiments, the anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes comparing at least one collection of object representations from the new stream of collections of object representations with at least one collection of object representations from the first stream of collections of object representations. In further embodiments, the comparing at least one collection of object representations from the new stream of collections of object representations with at least one collection of object representations from the first stream of collections of object representations includes comparing at least one object representation of the at least one collection of object representations from the new stream of collections of object representations with at least one object representation of the at least one collection of object representations from the first stream of collections of object representations. The comparing at least one object representation of the at least one collection of object representations from the new stream of collections of object representations with at least one object representation of the at least one collection of object representations from the first stream of collections of object representations may include comparing at least one object property of the at least one object representation of the at least one collection of object representations from the new stream of collections of object representations with at least one object property of the at least one object representation of the at least one collection of object representations from the first stream of collections of object representations. In further embodiments, the anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes comparing at least one object representation of the at least one collection of object representations from the new stream of collections of object representations with at least one object representation of the at least one collection of object representations from the first stream of collections of object representations. In further embodiments, the anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes comparing at least one object property of at least one object representation of at least one collection of object representations from the new stream of collections of object representations with at least one object property of at least one object representation of at least one collection of object representations from the first stream of collections of object representations.

In certain embodiments, the anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that there is at least a partial match between one or more portions of the new stream of collections of object representations and one or more portions of the first stream of collections of object representations. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that a similarity between at least one portion of the new stream of collections of object representations and at least one portion of the first stream of collections of object representations exceeds a similarity threshold. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining a substantial similarity between at least one portion of the new stream of collections of object representations and at least one portion of the first stream of collections of object representations. The substantial similarity may be achieved when a similarity between the at least one portion of the new stream of collections of object representations and the at least one portion of the first stream of collections of object representations exceeds a similarity threshold. The substantial similarity may be achieved when a number or a percentage of matching or partially matching portions of the new stream of collections of object representations and portions of the first stream of collections of object representations exceeds a threshold number or threshold percentage. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that a number or a percentage of matching or partially matching collections of object representations from the new stream of collections of object representations and from the first stream of collections of object representations exceeds a threshold number or threshold percentage. The matching or partially matching collections of object representations from the new stream of collections of object representations and from the first stream of collections of object representations may be determined factoring in at least one of: an importance of a collection of object representations, an order of a collection of object representations, a threshold for a similarity in a collection of object repre-

sentations, or a threshold for a difference in a collection of object representations. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that a number or a percentage of matching or partially matching object representations from the new stream of collections of object representations and from the first stream of collections of object representations exceeds a threshold number or threshold percentage. The matching or partially matching object representations from the new stream of collections of object representations and from the first stream of collections of object representations may be determined factoring in at least one of: an association of an object representation with a collection of object representations, a type of an object representation, an importance of an object representation, a threshold for a similarity in an object representation, or a threshold for a difference in an object representation. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that a number or a percentage of matching or partially matching object properties from the new stream of collections of object representations and from the first stream of collections of object representations exceeds a threshold number or threshold percentage. The matching or partially matching object properties from the new stream of collections of object representations and from the first stream of collections of object representations may be determined factoring in at least one of: an association of an object property with an object representation, an association of an object property with a collection of object representations, a category of an object property, an importance of an object property, a threshold for a similarity in an object property, or a threshold for a difference in an object property. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that there is at least a partial match between at least one collection of object representations from the new stream of collections of object representations and at least one collection of object representations from the first stream of collections of object representations. The determining that there is at least a partial match between at least one collection of object representations from the new stream of collections of object representations and at least one collection of object representations from the first stream of collections of object representations may include determining that there is at least a partial match between at least one object representation of the at least one collection of object representations from the new stream of collections of object representations and at least one object representation of the at least one collection of object representations from the first stream of collections of object representations. The determining that there is at least a partial match between at least one object representation of the at least one collection of object representations from the new stream of collections of object representations and at least one object representation of the at least one collection of object representations from the first stream of collections of object representations may include determining that there is at least a partial match between at least one object property of the at least one object representation of the at least one collection of object representations from the new stream of collections of object representations and at least one object property of the at least one object representation of the at least one collection of object representations from the first stream of collections of object representations.

In some embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations instead of or prior to an instruction set that would have been executed next. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying one or more instruction sets of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes inserting the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations into a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes redirecting the processor circuit to the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes redirecting the processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes transmitting, to the processor circuit for execution, the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes issuing an interrupt to the processor circuit and executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations following the interrupt. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes causing the application to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the causing the

processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying one or more instruction sets of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes redirecting the application to the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes redirecting the application to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes causing the avatar of the application to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying one or more instruction sets of the avatar of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying the avatar of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes redirecting the avatar of the application to the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes redirecting the avatar of the application to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying at least one of: the memory unit, a register of the processor circuit, a storage, or a repository where instruction sets are stored or used. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying at least one of: an element of the processor circuit, an element of the application, an element of the avatar of the application, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of the avatar of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes utilizing an assembly language. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation,

JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes adding or inserting additional code into a code of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes at least one of: modifying, removing, rewriting, or overwriting a code of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes adding or inserting additional code into a code of the avatar of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes at least one of: modifying, removing, rewriting, or overwriting a code of the avatar of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of the avatar of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes implementing a user's knowledge, style, or methodology of operating the avatar of the application in a circumstance. In further embodiments, the system further comprises: an interface configured to cause execution of instruction sets, wherein the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations by the processor circuit is caused by the interface. The interface may include a modification interface.

In certain embodiments, the avatar's performing the one or more operations defined by the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes implementing a user's knowledge, style, or methodology of operating the avatar of the application in a circumstance.

In some embodiments, the artificial intelligence unit is further configured to: receive at least one extra information. In further embodiments, the at least one extra information include one or more of: a time information, a location information, a computed information, a visual information, an acoustic information, or a contextual information. In further embodiments, the at least one extra information include one or more of: an information on the avatar of the application, an information on the avatar's circumstance, an information on an object, an information on an object representation, an information on a collection of object representations, an information on a stream of collections of object representations, an information on an instruction set, an information on the application, an information on the processor circuit, or an information on a user. In further embodiments, the artificial intelligence unit is further configured to: learn the first stream of collections of object representations correlated with the at least one extra information. The learning the first stream of collections of object representations correlated with at least one extra information may include correlating the first stream of collections of object representations with the at least one extra information. The learning the first stream of collections of object representations correlated with at least one extra information may include storing the first stream of collections of object representations correlated with the at least one extra information into the memory unit. In further embodiments, the anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based on at least a partial match between an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations. The anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based on at least a partial match between an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations may include comparing an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations. The anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based on at least a partial match between an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations may include determining that a similarity between an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations exceeds a similarity threshold.

In certain embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: cause the user interface to present a user with an option to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: receive, via the user interface, a user's selection to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In fur-

ther embodiments, the artificial intelligence unit is further configured to: rate the executed first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. The rating the executed first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations may include causing a user interface to display the executed first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations along with one or more rating values as options to be selected by a user. The rating the executed first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations may include rating the executed first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations without a user input. In further embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: cause the user interface to present a user with an option to cancel the execution of the executed first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the canceling the execution of the executed first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes restoring the processor circuit, the application, or the avatar of the application to a prior state. The restoring the processor circuit, the application, or the avatar of the application to a prior state may include saving the state of the processor circuit, the application, or the avatar of the application prior to executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations.

In some embodiments, the system further comprises: an input device configured to receive a user's operating directions, the user's operating directions for instructing the processor circuit, the application, or the avatar of the application on how to operate the avatar of the application.

In certain embodiments, the autonomous avatar operating includes a partially or a fully autonomous avatar operating. The partially autonomous avatar operating may include executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations responsive to a user confirmation. The fully autonomous avatar operating may include executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations without a user confirmation.

In some embodiments, the artificial intelligence unit is further configured to: receive a second stream of collections of object representations, the second stream of collections of object representations including one or more object representations representing one or more objects of the application; receive a second one or more instruction sets for operating the avatar of the application; and learn the second stream of collections of object representations correlated with the second one or more instruction sets for operating the avatar of the application. In further embodiments, the second stream of collections of object representations includes one or more collections of object representations, and wherein each collection of object representations of the second stream of collections of object representations includes one or more object representations. In further embodiments, the second stream of collections of object representations is received over a second time period. In further embodiments, the second stream of collections of object representations includes a unit of knowledge of the avatar's circumstance over a second time period. In further embodiments, the second stream of collections of object representations includes or is associated with a time stamp, an order, or a time related information. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application and the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the avatar of the application include creating a connection between the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the avatar of the application. The connection may include or is associated with at least one of: an occurrence count, a weight, a parameter, or a data. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application and the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the avatar of the application include updating a connection between the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the avatar of the application. In further embodiments, the updating the connection between the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the avatar of the application includes updating at least one of: an occurrence count, a weight, a parameter, or a data included in or associated with the connection. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application includes storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application into a first node of a knowledgebase, and wherein the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the avatar of the application includes storing the second stream of collections of object representations correlated with the second one or more instruction sets for operating the avatar of the application into a second node of the knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. The knowledgebase may be stored in the memory unit. The learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application and the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the avatar of the application may include creating a connection between the first node and the second node.

The learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application and the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the avatar of the application may include updating a connection between the first node and the second node. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application is stored into a first node of a neural network and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the avatar of the application is stored into a second node of the neural network. The first node and the second node may be connected by a connection. The first node may be part of a first layer of the neural network and the second node may be part of a second layer of the neural network. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application is stored into a first node of a graph and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the avatar of the application is stored into a second node of the graph. The first node and the second node may be connected by a connection. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application is stored into a first node of a sequence and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the avatar of the application is stored into a second node of the sequence.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first stream of collections of object representations, the first stream of collections of object representations including one or more object representations representing one or more objects of an application. The operations may further comprise: receiving a first one or more instruction sets for operating an avatar of the application. The operations may further comprise: learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application. The operations may further comprise: receiving a new stream of collections of object representations, the new stream of collections of object representations including one or more object representations representing one or more objects of the application. The operations may further comprise: anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The operations may further comprise: causing an execution of the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, wherein the avatar of the application performs one or more operations defined by the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations.

In certain embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes receiving the first one or more instruction sets for operating the avatar of the application from the one or more processor circuits or from another one or more processor circuits. In further embodiments, the execution of the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations is performed by the one or more processor circuits or by another one or more processor circuits.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first stream of collections of object representations by a processor circuit, the first stream of collections of object representations including one or more object representations representing one or more objects of an application. The method may further comprise: (b) receiving a first one or more instruction sets for operating an avatar of the application by the processor circuit. The method may further comprise: (c) learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application, the learning of (c) performed by the processor circuit. The method may further comprise: (d) receiving a new stream of collections of object representations by the processor circuit, the new stream of collections of object representations including one or more object representations representing one or more objects of the application. The method may further comprise: (e) anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, the anticipating of (e) performed by the processor circuit. The method may further comprise: (f) executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations, the executing of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the avatar of the application, one or more operations defined by the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations.

In some embodiments, the receiving of (b) includes receiving the first one or more instruction sets for operating the avatar of the application from the processor circuit or from another processor circuit. In further embodiments, the executing of (f) is performed by the processor circuit or by another processor circuit.

The aforementioned system, the non-transitory computer storage medium, and/or the method may include any elements, operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable as well as the following embodiments.

In certain embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application includes storing the first stream of collections of

object representations correlated with the first one or more instruction sets for operating the avatar of the application into the memory unit, the memory unit comprising a plurality of streams of collections of object representations correlated with one or more instruction sets for operating the avatar of the application.

In some embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations instead of or prior to an instruction set that would have been executed next. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying one or more instruction sets of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying a register or an element of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes inserting the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations into a register or an element of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes redirecting a processor circuit to the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes redirecting a processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes transmitting, to a processor circuit for execution, the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes issuing an interrupt to a processor circuit and executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations following the interrupt. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes causing the application to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying one or more instruction sets of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes redirecting the application to the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes redirecting the application to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes causing the avatar of the application to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying one or more instruction sets of the avatar of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying the avatar of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes redirecting the avatar of the application to the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes redirecting the avatar of the application to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying at least one of: a memory unit, a register of a processor circuit, a storage, or a repository where instruction sets are stored or used. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying at least one of: an element of a processor circuit, an element of the application, an element of the avatar of the application,

a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of the avatar of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes utilizing an assembly language. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes adding or inserting additional code into a code of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of

the application correlated with the first stream of collections of object representations includes at least one of: modifying, removing, rewriting, or overwriting a code of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes adding or inserting additional code into a code of the avatar of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes at least one of modifying, removing, rewriting, or overwriting a code of the avatar of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of the avatar of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes implementing a user's knowledge, style, or methodology of operating the avatar of the application in a circumstance. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations via an interface. The interface may include a modification interface.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving at least one extra information. In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: learning the first stream of collections of object representations correlated with the at least one extra information. In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: presenting, via a user interface, a user with an option to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving, via a user interface, a user's selection to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: rating the executed first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: presenting, via a user interface, a user with an option to cancel the execution of the executed first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In certain embodiments, the operations of the non-transitory computer storage medium and/or the

method further comprise: receiving, via an input device, a user's operating directions, the user's operating directions for instructing the processor circuit, the application, or the avatar of the application on how to operate the avatar of the application. In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving a second stream of collections of object representations, the second stream of collections of object representations including one or more object representations representing one or more objects of the application; receiving a second one or more instruction sets for operating the avatar of the application; and learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the avatar of the application.

The aforementioned system, the non-transitory computer storage medium, and/or the method may include any elements, operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable as well as the following embodiments.

In some aspects, the disclosure relates to a system for learning an avatar's circumstances for autonomous avatar operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets of an application. The system may further comprise: a memory unit configured to store data. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first stream of collections of object representations, the first stream of collections of object representations including one or more object representations representing one or more objects of the application. The artificial intelligence unit may be further configured to: receive a first one or more instruction sets for operating an avatar of the application. The artificial intelligence unit may be further configured to: learn the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first stream of collections of object representations, the first stream of collections of object representations including one or more object representations representing one or more objects of an application. The operations may further comprise: receiving a first one or more instruction sets for operating an avatar of the application. The operations may further comprise: learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first stream of collections of object representations by a processor circuit, the first stream of collections of object representations including one or more object representations representing one or more objects of an application. The method may further comprise: (b) receiving a first one or more instruction sets for operating an avatar of the application by the processor circuit. The method may further comprise: (c) learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application, the learning of (c) performed by the processor circuit.

The aforementioned system, the non-transitory computer storage medium, and/or the method may include any elements, operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable as well as the following embodiments.

In some aspects, the disclosure relates to a system for using an avatar's circumstances for autonomous avatar operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets of an application. The system may further comprise: a memory unit configured to store data. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: access the memory unit that comprises a plurality of streams of collections of object representations correlated with one or more instruction sets for operating an avatar of the application, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the avatar of the application including a first stream of collections of object representations correlated with a first one or more instruction sets for operating the avatar of the application. The artificial intelligence unit may be further configured to: receive a new stream of collections of object representations, the new stream of collections of object representations including one or more object representations representing one or more objects of the application. The artificial intelligence unit may be further configured to: anticipate the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the avatar of the application performs one or more operations defined by the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: accessing a memory unit that comprises a plurality of streams of collections of object representations correlated with one or more instruction sets for operating an avatar of an application, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the avatar of the application including a first stream of collections of object representations correlated with a first one or more instruction sets for operating the avatar of the application. The operations may further comprise: receiving a new stream of collections of object representations, the new stream of collections of object representations including one or more object representations representing one or more objects of the application. The operations may further comprise: anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based on at least a partial match between the

new stream of collections of object representations and the first stream of collections of object representations. The operations may further comprise: causing an execution of the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, wherein the avatar of the application performs one or more operations defined by the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations.

In some aspects, the disclosure relates to a method comprising: (a) accessing a memory unit that comprises a plurality of streams of collections of object representations correlated with one or more instruction sets for operating an avatar of an application, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the avatar of the application including a first stream of collections of object representations correlated with a first one or more instruction sets for operating the avatar of the application, the accessing of (a) performed by a processor circuit. The method may further comprise: (b) receiving a new stream of collections of object representations by the processor circuit, the new stream of collections of object representations including one or more object representations representing one or more objects of the application. The method may further comprise: (c) anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, the anticipating of (c) performed by the processor circuit. The method may further comprise: (d) executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations, the executing of (d) performed in response to the anticipating of (c). The method may further comprise: (e) performing, by the avatar of the application, one or more operations defined by the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations.

The aforementioned system, the non-transitory computer storage medium, and/or the method may include any elements, operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable as well as the following embodiments.

In some aspects, the disclosure relates to a system for learning and using an application's circumstances for autonomous application operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets of an application. The system may further comprise: a memory unit configured to store data. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first collection of object representations, the first collection of object representations including one or more object representations representing one or more objects of the application. The

artificial intelligence unit may be further configured to: receive a first one or more instruction sets for operating the application. The artificial intelligence unit may be further configured to: learn the first collection of object representations correlated with the first one or more instruction sets for operating the application. The artificial intelligence unit may be further configured to: receive a new collection of object representations, the new collection of object representations including one or more object representations representing one or more objects of the application. The artificial intelligence unit may be further configured to: anticipate the first one or more instruction sets for operating the application correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the first one or more instruction sets for operating the application correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the application performs one or more operations defined by the first one or more instruction sets for operating the application correlated with the first collection of object representations.

In certain embodiments, the first collection of object representations includes a unit of knowledge of the application's circumstance at a first time. In further embodiments, an application's circumstance includes one or more objects of the application.

In some embodiments, the first one or more instruction sets for operating the application include one or more instruction sets executed in operating the application. In further embodiments, the receiving the first one or more instruction sets for operating the application includes at least one of: tracing, profiling, or instrumentation of the application.

In certain embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the application include a unit of knowledge of how the application operated in a circumstance. In further embodiments, the leaning the first collection of object representations correlated with the first one or more instruction sets for operating the application includes learning a user's knowledge, style, or methodology of operating the application in a circumstance.

In some embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the application includes storing the first collection of object representations correlated with the first one or more instruction sets for operating the application into the memory unit, the memory unit comprising a plurality of collections of object representations correlated with one or more instruction sets for operating the application. The plurality of collections of object representations correlated with one or more instruction sets for operating the application may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure.

In certain embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the application correlated with the first collection of object representations includes causing the application to execute the first one or more instruction sets for operating the application correlated with the first collection of object representations. In further embodiments, the causing the

processor circuit to execute the first one or more instruction sets for operating the application correlated with the first collection of object representations includes implementing a user's knowledge, style, or methodology of operating the application in a circumstance.

In some embodiments, the artificial intelligence unit is further configured to: receive a second collection of object representations, the second collection of object representations including one or more object representations representing one or more objects of the application; receive a second one or more instruction sets for operating the application; and learn the second collection of object representations correlated with the second one or more instruction sets for operating the application. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the application includes storing the first collection of object representations correlated with the first one or more instruction sets for operating the application into a first node of a knowledgebase, and wherein the learning the second collection of object representations correlated with the second one or more instruction sets for operating the application includes storing the second collection of object representations correlated with the second one or more instruction sets for operating the application into a second node of the knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first collection of object representations, the first collection of object representations including one or more object representations representing one or more objects of an application. The operations may further comprise: receiving a first one or more instruction sets for operating the application. The operations may further comprise: learning the first collection of object representations correlated with the first one or more instruction sets for operating the application. The operations may further comprise: receiving a new collection of object representations, the new collection of object representations including one or more object representations representing one or more objects of the application. The operations may further comprise: anticipating the first one or more instruction sets for operating the application correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further comprise: causing an execution of the first one or more instruction sets for operating the application correlated with the first collection of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the application correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, wherein the application performs one or more operations defined by the first one or more instruction sets for operating the application correlated with the first collection of object representations.

In some embodiments, the receiving the first one or more instruction sets for operating the application includes receiving the first one or more instruction sets for operating the application from the one or more processor circuits or from another one or more processor circuits. In further embodiments, the execution of the first one or more instruction sets for operating the application correlated with the first collection of object representations is performed by the one or more processor circuits or by another one or more processor circuits.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a processor circuit, the first collection of object representations including one or more object representations representing one or more objects of an application. The method may further comprise: (b) receiving a first one or more instruction sets for operating an application by the processor circuit. The method may further comprise: (c) learning the first collection of object representations correlated with the first one or more instruction sets for operating the application, the learning of (c) performed by the processor circuit. The method may further comprise: (d) receiving a new collection of object representations by the processor circuit, the new collection of object representations including one or more object representations representing one or more objects of the application. The method may further comprise: (e) anticipating the first one or more instruction sets for operating the application correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (e) performed by the processor circuit. The method may further comprise: (f) executing the first one or more instruction sets for operating the application correlated with the first collection of object representations, the executing of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the application, one or more operations defined by the first one or more instruction sets for operating the application correlated with the first collection of object representations.

In certain embodiments, the receiving of (b) includes receiving the first one or more instruction sets for operating the application from the processor circuit or from another processor circuit. In further embodiments, the executing of (f) is performed by the processor circuit or by another processor circuit.

The aforementioned system, the non-transitory computer storage medium, and/or the method may include any elements, operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable as well as the following embodiments.

In some aspects, the disclosure relates to a system for learning and using an application's circumstances for autonomous application operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets of an application. The system may further comprise: a memory unit configured to store data. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first stream of collections of object representations, the first stream of collections of object representations including one or more object representations representing one or more objects of the application. The artificial intelligence unit may be further configured to: receive a first one or more instruction sets for operating the application. The artificial intelligence unit may be further configured to: learn the first stream of collections of object representations correlated with the first one or

more instruction sets for operating the application. The artificial intelligence unit may be further configured to: receive a new stream of collections of object representations, the new stream of collections of object representations including one or more object representations representing one or more objects of the application. The artificial intelligence unit may be further configured to: anticipate the first one or more instruction sets for operating the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the first one or more instruction sets for operating the application correlated with the first stream of collections of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the application performs one or more operations defined by the first one or more instruction sets for operating the application correlated with the first stream of collections of object representations.

In some embodiments, the first stream of collections of object representations includes a unit of knowledge of the application's circumstance over a first time period. In further embodiments, an application's circumstance includes one or more objects of the application.

In certain embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the application include a unit of knowledge of how the application operated in a circumstance. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the application includes learning a user's knowledge, style, or methodology of operating the application in a circumstance.

In some embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the application includes storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the application into the memory unit, the memory unit comprising a plurality of streams of collections of object representations correlated with one or more instruction sets for operating the application. The plurality of streams of collections of object representations correlated with one or more instruction sets for operating the application may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure.

In certain embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the application correlated with the first stream of collections of object representations includes causing the application to execute the first one or more instruction sets for operating the application correlated with the first stream of collections of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the application correlated with the first stream of collections of object representations includes implementing a user's knowledge, style, or methodology of operating the application in a circumstance.

In some embodiments, the artificial intelligence unit is further configured to: receive a second stream of collections of object representations, the second stream of collections of object representations including one or more object representations representing one or more objects of the application; receive a second one or more instruction sets for operating the application; and learn the second stream of collections of object representations correlated with the second one or more instruction sets for operating the application. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the application includes storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the application into a first node of a knowledgebase, and wherein the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the application includes storing the second stream of collections of object representations correlated with the second one or more instruction sets for operating the application into a second node of the knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first stream of collections of object representations, the first stream of collections of object representations including one or more object representations representing one or more objects of an application. The operations may further comprise: receiving a first one or more instruction sets for operating the application. The operations may further comprise: learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the application. The operations may further comprise: receiving a new stream of collections of object representations, the new stream of collections of object representations including one or more object representations representing one or more objects of the application. The operations may further comprise: anticipating the first one or more instruction sets for operating the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The operations may further comprise: causing an execution of the first one or more instruction sets for operating the application correlated with the first stream of collections of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, wherein the application performs one or more operations defined by the first one or more instruction sets for operating the application correlated with the first stream of collections of object representations.

In certain embodiments, the receiving the first one or more instruction sets for operating the application includes receiving the first one or more instruction sets for operating the application from the one or more processor circuits or from another one or more processor circuits. In further embodiments, the execution of the first one or more instruction sets for operating the application correlated with the first stream of collections of object representations is performed by the one or more processor circuits or by another one or more processor circuits.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first stream of collections of object representations by a processor circuit, the first stream of collections of object representations including one or more object representations representing one or more objects of an application. The method may further comprise: (b) receiving a first one or more instruction sets for operating the application by the processor circuit. The method may further comprise: (c) learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the application, the learning of (c) performed by the processor circuit. The method may further comprise: (d) receiving a new stream of collections of object representations by the processor circuit, the new stream of collections of object representations including one or more object representations representing one or more objects of the application. The method may further comprise: (e) anticipating the first one or more instruction sets for operating the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, the anticipating of (e) performed by the processor circuit. The method may further comprise: (f) executing the first one or more instruction sets for operating the application correlated with the first stream of collections of object representations, the executing of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the application, one or more operations defined by the first one or more instruction sets for operating the application correlated with the first stream of collections of object representations.

In some embodiments, the receiving of (b) includes receiving the first one or more instruction sets for operating the application from the processor circuit or from another processor circuit. In further embodiments, the executing of (f) is performed by the processor circuit or by another processor circuit.

The aforementioned system, the non-transitory computer storage medium, and/or the method may include any elements, operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable as well as the following embodiments.

Other features and advantages of the disclosure will become apparent from the following description, including the claims and drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** illustrates a block diagram of Computing Device **70** that can provide processing capabilities used in some of the disclosed embodiments.

FIG. **2** illustrates an embodiment of Computing Device **70** comprising Unit for Learning and/or Using an Avatar's Circumstances for Autonomous Avatar Operation (ACAAO Unit **100**).

FIG. **3** illustrates an embodiment of utilizing Picture Renderer **91** and Picture Recognizer **92**.

FIG. **4** illustrates an embodiment of utilizing Sound Renderer **96** and Sound Recognizer **97**.

FIGS. **5**A-**5**B, illustrate an exemplary embodiment of Objects **615** in Avatar's **605** surrounding, and resulting Collection of Object Representations **525**

FIG. **6** illustrates some embodiments of obtaining instruction sets, data, and/or other information through tracing, profiling, or sampling of Processor **11** registers, memory, or other computing system components.

FIGS. **7**A-**7**E illustrate some embodiments of Instruction Sets **526**.

FIGS. **8**A-**8**B illustrate some embodiments of Extra Information **527**.

FIG. **9** illustrates an embodiment where ACAAO Unit **100** is part of or operating on Processor **11**.

FIG. **10** illustrates an embodiment where ACAAO Unit **100** resides on Server **96** accessible over Network **95**.

FIG. **11** illustrates an embodiment of Artificial Intelligence Unit **110**.

FIG. **12** illustrates an embodiment of Knowledge Structuring Unit **520** correlating individual Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527**.

FIG. **13** illustrates another embodiment of Knowledge Structuring Unit **520** correlating individual Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527**.

FIG. **14** illustrates an embodiment of Knowledge Structuring Unit **520** correlating streams of Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527**.

FIG. **15** illustrates another embodiment of Knowledge Structuring Unit **520** correlating streams of Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527**.

FIG. **16** illustrates various artificial intelligence methods, systems, and/or models that can be utilized in ACAAO Unit **100** embodiments.

FIGS. **17**A-**17**C illustrate embodiments of interconnected Knowledge Cells **800** and updating weights of Connections **853**.

FIG. **18** illustrates an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Collection of Knowledge Cells **530***d*.

FIG. **19** illustrates an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Neural Network **530***a*.

FIG. **20** illustrates an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Neural Network **530***a* comprising shortcut Connections **853**.

FIG. **21** illustrates an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Graph **530***b*.

FIG. **22** illustrates an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Collection of Sequences **530***c*.

FIG. **23** illustrates an embodiment of determining anticipatory Instruction Sets **526** from a single Knowledge Cell **800**.

FIG. **24** illustrates an embodiment of determining anticipatory Instruction Sets **526** by traversing a single Knowledge Cell **800**.

FIG. **25** illustrates an embodiment of determining anticipatory Instruction Sets **526** using collective similarity comparisons.

FIG. **26** illustrates an embodiment of determining anticipatory Instruction Sets **526** using Neural Network **530***a*.

FIG. **27** illustrates an embodiment of determining anticipatory Instruction Sets **526** using Graph **530***b*.

FIG. **28** illustrates an embodiment of determining anticipatory Instruction Sets **526** using Collection of Sequences **530***c*.

FIG. **29** illustrates some embodiments of modifying execution and/or functionality of Avatar **605** and/or Application Program **18** through modification of Processor **11** registers, memory, or other computing system components.

FIG. **30** illustrates a flow chart diagram of an embodiment of method **9100** for learning and/or using an avatar's circumstances for autonomous avatar operation.

FIG. **31** illustrates a flow chart diagram of an embodiment of method **9200** for learning and/or using an avatar's circumstances for autonomous avatar operation.

FIG. **32** illustrates a flow chart diagram of an embodiment of method **9300** for learning and/or using an avatar's circumstances for autonomous avatar operation.

FIG. **33** illustrates a flow chart diagram of an embodiment of method **9400** for learning and/or using an application's circumstances for autonomous application operation.

FIG. **34** illustrates a flow chart diagram of an embodiment of method **9500** for learning and/or using an application's circumstances for autonomous application operation.

FIG. **35** illustrates a flow chart diagram of an embodiment of method **9600** for learning and/or using an application's circumstances for autonomous application operation.

FIG. **36** illustrates an exemplary embodiment of Soldier **605***a* within 3D Computer Game **18***a*.

FIG. **37** illustrates an exemplary embodiment of Tank **605***b* within 2D Computer Game **18***b*.

FIG. **38** illustrates an exemplary embodiment of utilizing Area of Interest **450** around Tank **605***b*.

FIG. **39** illustrates an exemplary embodiment of multiple Avatars **605** within Computer Game **18***c*.

Like reference numerals in different figures indicate like elements. Horizontal or vertical " . . . " or other such indicia may be used to indicate additional instances of the same type of element. n, m, x, or other such letters or indicia represent integers or other sequential numbers that follow the sequence where they are indicated. It should be noted that n, m, x, or other such letters or indicia may represent different numbers in different elements even where the elements are depicted in the same figure. In general, n, m, x, or other such letters or indicia may follow the sequence and/or context where they are indicated. Any of these or other such letters or indicia may be used interchangeably depending on context and space available. The drawings are not necessarily to scale, with emphasis instead being placed upon illustrating the embodiments, principles, and concepts of the disclosure. A line or arrow between any of the disclosed elements comprises an interface that enables the coupling, connection, and/or interaction between the elements.

DETAILED DESCRIPTION

The disclosed artificially intelligent devices, systems, and methods for learning and/or using an avatar's circumstances for autonomous avatar operation comprise apparatuses, systems, methods, features, functionalities, and/or applications that enable learning an avatar's circumstances including objects with various properties along with correlated instruction sets for operating the avatar, storing this knowledge in a knowledgebase (i.e. neural network, graph, sequences, etc.), and/or operating an avatar autonomously. The disclosed artificially intelligent devices, systems, and methods for learning and/or using an avatar's circumstances for autonomous avatar operation, any of their elements, any of their embodiments, or a combination thereof can generally be referred to as ACAAO, ACAAO Unit, or as other suitable name or reference.

Referring now to FIG. **1**, an embodiment is illustrated of Computing Device **70** (also referred to as simply as computing device, computing system, or other suitable name or reference, etc.) that can provide processing capabilities used in some embodiments of the forthcoming disclosure. Later described devices, systems, and methods, in combination with processing capabilities of Computing Device **70**, enable learning and/or using an avatar's circumstances for autonomous avatar operation and/or other functionalities described herein. Various embodiments of the disclosed devices, systems, and methods include hardware, functions, logic, programs, and/or a combination thereof that can be implemented using any type or form of computing, computing enabled, or other device or system such as a mobile device, a computer, a computing enabled telephone, a server, a gaming device, a television device, a digital camera, a GPS receiver, a media player, an embedded device, a supercomputer, a wearable device, an implantable device, a cloud, or any other type or form of computing, computing enabled, or other device or system capable of performing the operations described herein.

In some designs, Computing Device **70** comprises hardware, processing techniques or capabilities, programs, or a combination thereof. Computing Device **70** includes one or more central processing units, which may also be referred to as processors **11**. Processor **11** includes one or more memory ports **10** and/or one or more input-output ports, also referred to as I/O ports **15**, such as I/O ports **15**A and **15**B. Processor **11** may be special or general purpose. Computing Device **70** may further include memory **12**, which can be connected to the remainder of the components of Computing Device **70** via bus **5**. Memory **12** can be connected to processor **11** via memory port **10**. Computing Device **70** may also include display device **21** such as a monitor, projector, glasses, and/or other display device. Computing Device **70** may also include Human-machine Interface **23** such as a keyboard, a pointing device, a mouse, a touchscreen, a joystick, a remote controller, and/or other input device. In some implementations, Human-machine Interface **23** can be connected with bus **5** or directly connected with specific elements of Computing Device **70**. Computing Device **70** may include additional elements such as one or more input/output devices **13**. Processor **11** may include or be interfaced with cache memory **14**. Storage **27** may include memory, which provides an operating system **17** (i.e. also referred to as OS **17**, etc.), additional application programs **18**, and/or data space **19** in which additional data or information can be stored. Alternative memory device **16** can be connected to the remaining components of Computing Device **70** via bus **5**. Network interface **25** can also be connected with bus **5** and be used to communicate with external computing devices via a network. Some or all described elements of Computing Device **70** can be directly or operatively connected or coupled with each other using any other connection means known in art. Other additional elements may be included as needed, or some of the disclosed ones may be excluded, or a combination thereof may be utilized in alternate implementations of Computing Device **70**.

Processor **11** includes one or more circuits or devices that can execute instructions fetched from memory **12** and/or other element. Processor **11** may include any combination of hardware and/or processing techniques or capabilities for executing or implementing logic functions or programs.

Processor **11** may include a single core or a multi core processor. Processor **11** includes the functionality for loading operating system **17** and operating any application programs **18** thereon. In some embodiments, Processor **11** can be provided in a microprocessing or a processing unit, such as, for example, Snapdragon processor produced by Qualcomm Inc., processor by Intel Corporation of Mountain View, Calif., processor manufactured by Motorola Corporation of Schaumburg, Ill.; processor manufactured by Transmeta Corporation of Santa Clara, Calif.; processor manufactured by International Business Machines of White Plains, N.Y.; processor manufactured by Advanced Micro Devices of Sunnyvale, Calif., or any computing circuit or device for performing similar functions. In other embodiments, processor **11** can be provided in a graphics processing unit (GPU), visual processing unit (VPU), or other highly parallel processing circuit or device such as, for example, nVidia GeForce line of GPUs, AMD Radeon line of GPUs, and/or others. Such GPUs or other highly parallel processing circuits or devices may provide superior performance in processing operations on neural networks, graphs, and/or other data structures. In further embodiments, processor **11** can be provided in a micro controller such as, for example, Texas instruments, Atmel, Microchip Technology, ARM, Silicon Labs, Intel, and/or other lines of micro controllers. In further embodiments, processor **11** can be provided in a quantum processor such as, for example, D-Wave Systems, Microsoft, Intel, IBM, Google, Toshiba, and/or other lines of quantum processors. In further embodiments, processor **11** can be provided in a biocomputer such as DNA-based computer, protein-based computer, molecule-based computer, and/or others. In further embodiments, processor **11** includes any circuit or device for performing logic operations. Processor **11** can be based on any of the aforementioned or other available processors capable of operating as described herein.

Computing Device **70** may include one or more of the aforementioned or other processors. In some designs, processor **11** can communicate with memory **12** via a system bus **5**. In other designs, processor **11** can communicate directly with memory **12** via a memory port **10**.

Memory **12** includes one or more circuits or devices capable of storing data. In some embodiments, Memory **12** can be provided in a semiconductor or electronic memory chip such as static random access memory (SRAM), Flash memory, Burst SRAM or SynchBurst SRAM (BSRAM), Dynamic random access memory (DRAM), Fast Page Mode DRAM (FPM DRAM), Enhanced DRAM (EDRAM), Extended Data Output RAM (EDO RAM), Extended Data Output DRAM (EDO DRAM), Burst Extended Data Output DRAM (BEDO DRAM), Enhanced DRAM (EDRAM), synchronous DRAM (SDRAM), JEDEC SRAM, PC100 SDRAM, Double Data Rate SDRAM (DDR SDRAM), Enhanced SDRAM (ESDRAM), SyncLink DRAM (SLDRAM), Direct Rambus DRAM (DRDRAM), Ferroelectric RAM (FRAM), and/or others. In other embodiments, Memory **12** includes any volatile memory. In general, Memory **12** can be based on any of the aforementioned or other available memories capable of operating as described herein.

Storage **27** includes one or more devices or mediums capable of storing data. In some embodiments, Storage **27** can be provided in a device or medium such as a hard drive, flash drive, optical disk, and/or others. In other embodiments, Storage **27** can be provided in a biological storage device such as DNA-based storage device, protein-based storage device, molecule-based storage device, and/or others. In further embodiments, Storage **27** can be provided in an optical storage device such as holographic storage, and/or others. In further embodiments, Storage **27** may include any non-volatile memory. In general, Storage **27** can be based on any of the aforementioned or other available storage devices or mediums capable of operating as described herein. In some aspects, Storage **27** may include any features, functionalities, and embodiments of Memory **12**, and vice versa, as applicable.

Processor **11** can communicate directly with cache memory **14** via a connection means such as a secondary bus which may also sometimes be referred to as a backside bus. In some embodiments, processor **11** can communicate with cache memory **14** using the system bus **5**. Cache memory **14** may typically have a faster response time than main memory **12** and can include a type of memory which is considered faster than main memory **12** such as, for example, SRAM, BSRAM, or EDRAM. Cache memory includes any structure such as multilevel caches, for example. In some embodiments, processor **11** can communicate with one or more I/O devices **13** via a system bus **5**. Various busses can be used to connect processor **11** to any of the I/O devices **13** such as a VESA VL bus, an ISA bus, an EISA bus, a MicroChannel Architecture (MCA) bus, a PCI bus, a PCI-X bus, a PCI-Express bus, a NuBus, and/or others. In some embodiments, processor **11** can communicate directly with I/O device **13** via HyperTransport, Rapid I/O, or InfiniBand. In further embodiments, local busses and direct communication can be mixed. For example, processor **11** can communicate with an I/O device **13** using a local interconnect bus and communicate with another I/O device **13** directly. Similar configurations can be used for any other components described herein.

Computing Device **70** may further include alternative memory such as a SD memory slot, a USB memory stick, an optical drive such as a CD-ROM drive, a CD-R/RW drive, a DVD-ROM drive or a BlueRay disc, a hard-drive, and/or any other device comprising non-volatile memory suitable for storing data or installing application programs. Computing Device **70** may further include a storage device **27** comprising any type or form of non-volatile memory for storing an operating system (OS) such as any type or form of Windows OS, Mac OS, Unix OS, Linux OS, Android OS, iPhone OS, mobile version of Windows OS, an embedded OS, or any other OS that can operate on Computing Device **70**. Computing Device **70** may also include application programs **18**, and/or data space **19** for storing additional data or information. In some embodiments, alternative memory **16** can be used as or similar to storage device **27**. Additionally, OS **17** and/or application programs **18** can be operable from a bootable medium such as, for example, a flash drive, a micro SD card, a bootable CD or DVD, and/or other bootable medium.

Application Program **18** (also referred to as program, computer program, application, script, code, or other suitable name or reference) comprises instructions that can provide functionality when executed by processor **11**. Application program **18** can be implemented in a high-level procedural or object-oriented programming language, or in a low-level machine or assembly language. Any language used can be compiled, interpreted, or translated into machine language. Application program **18** can be deployed in any form including as a stand-alone program or as a module, component, subroutine, or other unit suitable for use in a computing system. Application program **18** does not necessarily correspond to a file in a file system. A program can be stored in a portion of a file that may hold other programs or data, in a single file dedicated to the program, or in multiple

files (i.e. files that store one or more modules, sub programs, or portions of code, etc.). Application Program **18** can be delivered in various forms such as, for example, executable file, library, script, plugin, addon, applet, interface, console application, web application, application service provider (ASP)-type application, operating system, and/or other forms. Application program **18** can be deployed to be executed on one computing device or on multiple computing devices (i.e. cloud, distributed, or parallel computing, etc.), or at one site or distributed across multiple sites interconnected by a network or an interface. Examples of Application Program **18** include a computer game, a virtual world application, a graphics application, a media application, a word processing application, a spreadsheet application, a database application, a web browser, a forms-based application, a global positioning system (GPS) application, a 2D application, a 3D application, an operating system, a factory automation application, a device control application, a vehicle control application, and/or other application or program.

Network interface **25** can be utilized for interfacing Computing Device **70** with other devices via a network through a variety of connections including telephone lines, wired or wireless connections, LAN or WAN links (i.e. 802.11, T1, T3, 56 kb, X.25, etc.), broadband connections (i.e. ISDN, Frame Relay, ATM, etc.), or a combination thereof. Examples of networks include the Internet, an intranet, an extranet, a local area network (LAN), a wide area network (WAN), a personal area network (PAN), a home area network (HAN), a campus area network (CAN), a metropolitan area network (MAN), a global area network (GAN), a storage area network (SAN), virtual network, a virtual private network (VPN), a Bluetooth network, a wireless network, a wireless LAN, a radio network, a HomePNA, a power line communication network, a G.hn network, an optical fiber network, an Ethernet network, an active networking network, a client-server network, a peer-to-peer network, a bus network, a star network, a ring network, a mesh network, a star-bus network, a tree network, a hierarchical topology network, and/or other networks. Network interface **25** may include a built-in network adapter, network interface card, PCMCIA network card, card bus network adapter, wireless network adapter, Bluetooth network adapter, WiFi network adapter, USB network adapter, modem, and/or any other device suitable for interfacing Computing Device **70** with any type of network capable of communication and/or operations described herein.

I/O devices **13** may be present in various shapes or forms in Computing Device **70**. Examples of I/O device **13** capable of input include a joystick, a keyboard, a mouse, a trackpad, a trackpoint, a trackball, a microphone, a drawing tablet, a glove, a tactile input device, a still or video camera, and/or other input device. Examples of I/O device **13** capable of output include a video display, a projector, a glasses, a speaker, a tactile output device, and/or other output device. Examples of I/O device **13** capable of input and output include a touchscreen, a disk drive, an optical storage device, a modem, a network card, and/or other input/output device. I/O device **13** can be interfaced with processor **11** via an I/O port **15**, for example. In some aspects, I/O device **13** can be a bridge between system bus **5** and an external communication bus such as a USB bus, an Apple Desktop Bus, an RS-232 serial connection, a SCSI bus, a FireWire bus, a FireWire 800 bus, an Ethernet bus, an AppleTalk bus, a Gigabit Ethernet bus, an Asynchronous Transfer Mode bus, a HIPPI bus, a Super HIPPI bus, a SerialPlus bus, a SCI/LAMP bus, a FibreChannel bus, a Serial Attached small computer system interface bus, and/or other bus.

An output interface (not shown) such as a graphical output interface, an acoustic output interface, a tactile output interface, a renderer, any device driver (i.e. audio, video, or other driver), and/or other output interface or system can be utilized to process output from elements of Computing Device **70** for conveyance on an output device such as Display **21**. In some aspects, Display **21** or other output device itself may include an output interface for processing output from elements of Computing Device **70**. Further, an input interface (not shown) such as a keyboard listener, a touchscreen listener, a mouse listener, any device driver (i.e. audio, video, keyboard, mouse, touchscreen, or other driver), and/or other input interface or system can be utilized to process input from Human-machine Interface **23** or other input device for use by elements of Computing Device **70**. In some aspects, Human-machine Interface **23** or other input device itself may include an input interface for processing input for use by elements of Computing Device **70**.

Computing Device **70** may include or be connected to multiple display devices **21**. Display devices **21** can each be of the same or different type or form. Computing Device **70** and/or its elements comprise any type or form of suitable hardware, programs, or a combination thereof to support, enable, or provide for the connection and use of multiple display devices **21**. In one example, Computing Device **70** includes any type or form of video adapter, video card, driver, and/or library to interface, communicate, connect, or otherwise use display devices **21**. In some aspects, a video adapter may include multiple connectors to interface to multiple display devices **21**. In other aspects, Computing Device **70** includes multiple video adapters, with each video adapter connected to one or more display devices **21**. In some embodiments, Computing Device's **70** operating system can be configured for using multiple displays **21**. In other embodiments, one or more display devices **21** can be provided by one or more other computing devices such as remote computing devices connected to Computing Device **70** via a network or an interface.

Computing Device **70** can operate under the control of operating system **17**, which may support Computing Device's **70** basic functions, interface with and manage hardware resources, interface with and manage peripherals, provide common services for application programs, schedule tasks, and/or perform other functionalities. A modem operating system enables features and functionalities such as a high resolution display, graphical user interface (GUI), touchscreen, cellular network connectivity (i.e. mobile operating system, etc.), Bluetooth connectivity, WiFi connectivity, global positioning system (GPS) capabilities, mobile navigation, microphone, speaker, still picture camera, video camera, voice recorder, speech recognition, music player, video player, near field communication, personal digital assistant (PDA), and/or other features, functionalities, or applications. For example, Computing Device **70** can use any conventional operating system, any embedded operating system, any real-time operating system, any open source operating system, any video gaming operating system, any proprietary operating system, any online operating system, any operating system for mobile computing devices, or any other operating system capable of running on Computing Device **70** and performing operations described herein. Example of operating systems include Windows XP, Windows 7, Windows 8, Windows 10, etc. manufactured by Microsoft Corporation of Redmond, Wash.; Mac OS, iPhone OS, etc. manufactured by Apple Computer of Cupertino,

Calif.; OS/2 manufactured by International Business Machines of Armonk, N.Y.; Linux, a freely-available operating system distributed by Caldera Corp. of Salt Lake City, Utah; or any type or form of a Unix operating system, and/or others. Any operating systems such as the ones for Android devices can similarly be utilized.

Computing Device **70** can be implemented as or be part of various model architectures such as web services, distributed computing, grid computing, cloud computing, and/or other architectures. For example, in addition to the traditional desktop, server, or mobile operating system architectures, a cloud-based operating system can be utilized to provide the structure on which embodiments of the disclosure can be implemented. Other aspects of Computing Device **70** can also be implemented in the cloud without departing from the spirit and scope of the disclosure. For example, memory, storage, processing, and/or other elements can be hosted in the cloud. In some embodiments, Computing Device **70** can be implemented on multiple devices. For example, a portion of Computing Device **70** can be implemented on a mobile device and another portion can be implemented on wearable electronics.

Computing Device **70** can be or include any mobile device, a mobile phone, a smartphone (i.e. iPhone, Windows phone, Blackberry phone, Android phone, etc.), a tablet, a personal digital assistant (PDA), wearable electronics, implantable electronics, and/or other mobile device capable of implementing the functionalities described herein. Computing Device **70** can also be or include an embedded device, which can be any device or system with a dedicated function within another device or system. Embedded systems range from the simplest ones dedicated to one task with no user interface to complex ones with advanced user interface that may resemble modem desktop computer systems. Examples of devices comprising an embedded device include a mobile telephone, a personal digital assistant (PDA), a gaming device, a media player, a digital still or video camera, a pager, a television device, a set-top box, a personal navigation device, a global positioning system (GPS) receiver, a portable storage device (i.e. a USB flash drive, etc.), a digital watch, a DVD player, a printer, a microwave oven, a washing machine, a dishwasher, a gateway, a router, a hub, an automobile entertainment system, an automobile navigation system, a refrigerator, a washing machine, a factory automation device, an assembly line device, a factory floor monitoring device, a thermostat, an automobile, a factory controller, a telephone, a network bridge, and/or other devices. An embedded device can operate under the control of an operating system for embedded devices such as MicroC/OS-II, QNX, VxWorks, eCos, TinyOS, Windows Embedded, Embedded Linux, and/or other embedded device operating systems.

Various implementations of the disclosed devices, systems, and methods can be realized in digital electronic circuitry, integrated circuitry, logic gates, application specific integrated circuits (ASICs), field programmable gate arrays (FPGAs), computer hardware, firmware, programs, virtual machines, and/or combinations thereof including their structural, logical, and/or physical equivalents.

The disclosed devices, systems, and methods may include clients and servers. A client and server are generally, but not always, remote from each other and typically, but not always, interact via a network or an interface. The relationship of a client and server may arise by virtue of computer programs running on their respective computers and having a client-server relationship to each other, for example.

The disclosed devices, systems, and methods can be implemented in a computing system that includes a back end component, a middleware component, a front end component, or any combination thereof. The components of the system can be interconnected by any form or medium of digital data communication such as, for example, a network.

Computing Device **70** may include or be interfaced with a computer program product comprising instructions or logic encoded on a computer-readable medium. Such instructions or logic, when executed, may configure or cause one or more processors **11** to perform the operations and/or functionalities disclosed herein. For example, a computer program can be provided or encoded on a computer-readable medium such as an optical medium (i.e. DVD-ROM, etc.), flash drive, hard drive, any memory, firmware, or other medium. Machine-readable medium, computer-readable medium, or other such terms may refer to any computer program product, apparatus, and/or device for providing instructions and/or data to one or more programmable processors. As such, machine-readable medium includes any medium that can send and/or receive machine instructions as a machine-readable signal. Examples of a machine-readable medium include a volatile and/or non-volatile medium, a removable and/or non-removable medium, a communication medium, a storage medium, and/or other medium. A non-transitory machine-readable medium comprises all machine-readable media except for a transitory, propagating signal.

In some embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using an avatar's circumstances for autonomous avatar operation, or elements thereof, can be implemented entirely or in part in a device (i.e. microchip, circuitry, logic gates, electronic device, computing device, special or general purpose processor, etc.) or system that comprises (i.e. hard coded, internally stored, etc.) or is provided with (i.e. externally stored, etc.) instructions for implementing ACAAO functionalities. As such, the disclosed artificially intelligent devices, systems, and methods for learning and/or using an avatar's circumstances for autonomous avatar operation, or elements thereof, may include the processing, memory, storage, and/or other features, functionalities, and embodiments of Computing Device **70** or elements thereof. Such device or system can operate on its own (i.e. standalone device or system, etc.), be embedded in another device or system (i.e. an industrial machine, a robot, a vehicle, a toy, a smartphone, a television device, an appliance, and/or any other device or system capable of housing the elements needed for ACAAO functionalities), work in combination with other devices or systems, or be available in any other configuration. In other embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using an avatar's circumstances for autonomous avatar operation, or elements thereof, may include or be interfaced with Alternative Memory **16** that provides instructions for implementing ACAAO functionalities to one or more Processors **11**. In further embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using an avatar's circumstances for autonomous avatar operation, or elements thereof, can be implemented entirely or in part as a computer program and executed by one or more Processors **11**. Such program can be implemented in one or more modules or units of a single or multiple computer programs. Such program may be able to attach to or interface with, inspect, and/or take control of another application program to implement ACAAO functionalities. In further embodiments, the disclosed artificially

intelligent devices, systems, and methods for learning and/or using an avatar's circumstances for autonomous avatar operation, or elements thereof, can be implemented as a network, web, distributed, cloud, or other such application accessed on one or more remote computing devices (i.e. servers, cloud, etc.) via Network Interface **25**, such remote computing devices including processing capabilities and instructions for implementing ACAAO functionalities. In further embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using an avatar's circumstances for autonomous avatar operation, or elements thereof, can be (1) attached to or interfaced with any computing device or application program, (2) included as a feature of an operating system, (3) built (i.e. hard coded, etc.) into any computing device or application program, and/or (4) available in any other configuration to provide its functionalities.

In some embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using an avatar's circumstances for autonomous avatar operation, or elements thereof, can be implemented at least in part in a computer program such as Java application or program. Java provides a robust and flexible environment for application programs including flexible user interfaces, robust security, built-in network protocols, powerful application programming interfaces, database or DBMS connectivity and interfacing functionalities, file manipulation capabilities, support for networked applications, and/or other features or functionalities. Application programs based on Java can be portable across many devices, yet leverage each device's native capabilities. Java supports the feature sets of most smartphones and a broad range of connected devices while still fitting within their resource constraints. Various Java platforms include virtual machine features comprising a runtime environment for application programs. Java platforms provide a wide range of user-level functionalities that can be implemented in application programs such as displaying text and graphics, playing and recording audio content, displaying and recording visual content, communicating with another computing device, and/or other functionalities. It should be understood that the disclosed artificially intelligent devices, systems, and methods for learning and/or using an avatar's circumstances for autonomous avatar operation, or elements thereof, are programming language, platform, and operating system independent. Examples of programming languages that can be used instead of or in addition to Java include C, C++, Cobol, Python, Java Script, Td, Visual Basic, Pascal, VB Script, Perl, PHP, Ruby, and/or other programming languages capable of implementing the functionalities described herein.

Where a reference to a specific ile or file type is used herein, other files or file types can be substituted.

Where a reference to a data structure is used herein, it should be understood that any variety of data structures can be used such as, for example, array, list, linked list, doubly linked list, queue, tree, heap, graph, map, grid, matrix, multi-dimensional matrix, table, database, database management system (DBMS), file, neural network, and/or any other type or form of a data structure including a custom one. A data structure may include one or more fields or data fields that are part of or associated with the data structure. A field or data field may include a data, an object, a data structure, and/or any other element or a reference/pointer thereto. A data structure can be stored in one or more memories, files, or other repositories. A data structure and/or elements thereof, when stored in a memory, file, or other repository, may be stored in a different arrangement than the arrangement of the data structure and/or elements thereof. For example, a sequence of elements can be stored in an arrangement other than a sequence in a memory, file, or other repository.

Where a reference to a repository is used herein, it should be understood that a repository may be or include one or more files or file systems, one or more storage locations or structures, one or more storage systems, one or more memory locations or structures, and/or other file, storage, memory, or data arrangements.

Where a reference to an interface is used herein, it should be understood that the interface comprises any hardware, device, system, program, method, and/or combination thereof that enable direct or operative coupling, connection, and/or interaction of the elements between which the interface is indicated. A line or arrow shown in the figures between any of the depicted elements comprises such interface. Examples of an interface include a direct connection, an operative connection, a wired connection (i.e. wire, cable, etc.), a wireless connection, a device, a network, a bus, a circuit, a firmware, a driver, a bridge, a program, a combination thereof, and/or others.

Where a reference to an element coupled or connected to another element is used herein, it should be understood that the element may be in communication or other interactive relationship with the other element.

Furthermore, an element coupled or connected to another element can be coupled or connected to any other element in alternate implementations. Terms coupled, connected, interfaced, or other such terms may be used interchangeably herein depending on context.

Where a reference to an element matching another element is used herein, it should be understood that the element may be equivalent or similar to the other element. Therefore, the term match or matching can refer to total equivalence or similarity depending on context.

Where a reference to a device is used herein, it should be understood that the device may include or be referred to as a system, and vice versa depending on context, since a device may include a system of elements and a system may be embodied in a device.

Where a reference to a collection of elements is used herein, it should be understood that the collection of elements may include one or more elements. In some aspects or contexts, a reference to a collection of elements does not imply that the collection is an element itself.

Where a mention of a function, method, routine, subroutine, or other such procedure is used herein, it should be understood that the function, method, routine, subroutine, or other such procedure comprises a call, reference, or pointer to the function, method, routine, subroutine, or other such procedure.

Where a mention of data, object, data structure, item, element, or thing is used herein, it should be understood that the data, object, data structure, item, element, or thing comprises a reference or pointer to the data, object, data structure, item, element, or thing.

Referring to FIG. **2**, an embodiment of Computing Device **70** comprising Unit for Learning and/or Using an Avatar's Circumstances for Autonomous Avatar Operation (ACAAO Unit **100**) is illustrated. Computing Device **70** also comprises interconnected Processor **11**, Display **21**, Human-machine Interface **23**, Memory **12**, and Storage **27**. Processor **11** includes or executes Application Program **18** comprising Avatar **605** and/or one or more Objects **615**. ACAAO Unit **100** comprises interconnected Artificial Intel-

ligence Unit **110**, Acquisition Interface **120**, Modification Interface **130**, and Object Processing Unit **140**. Other additional elements can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate embodiments.

In one example, the teaching presented by the disclosure can be implemented in a device or system for learning and/or using an avatar's circumstances for autonomous avatar operating. The device or system may include a processor circuit (i.e. Processor **11**, etc.) configured to execute instruction sets (i.e. Instruction Sets **526**, etc.) of an application. The device or system may further include a memory unit (i.e. Memory **12**, etc.) configured to store data. The device or system may further include an artificial intelligence unit (i.e. Artificial Intelligence Unit **110**, etc.). The artificial intelligence unit may be configured to receive a first collection of object representations (i.e. Collection of Object Representations **525**, etc.), the first collection of object representations including one or more object representations representing one or more objects of the application. The artificial intelligence unit may also be configured to receive a first one or more instruction sets for operating an avatar of the application. The artificial intelligence unit may also be configured to learn the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application. The artificial intelligence unit may also be configured to receive a new collection of object representations, the new collection of object representations including one or more object representations representing one or more objects of the application. The artificial intelligence unit may also be configured to anticipate the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may also be configured to cause the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the avatar of the application performs one or more operations defined by the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. Any of the operations of the aforementioned elements can be performed repeatedly and/or in different orders in alternate embodiments. In some embodiments, a collection of object representations may include or be substituted with a stream of collections of object representations. In some embodiments of applications that do not comprise an avatar or rely on avatar for their operation, the teaching presented by the disclosure can be implemented in a device or system for learning and/or using an application's circumstances for autonomous application operating. In such embodiments, an instruction set for operating an avatar of an application may include or be substituted with an instruction set for operating an application. Other additional elements can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate embodiments. The disclosed devices and systems may include any actions or operations of any of the disclosed methods such as methods **9100**, **9200**, **9300**, **9400**, **9500**, **9600**, and/or others (all later described).

User **50** (also referred to as simply as user or other suitable name or reference) comprises a human user or non-human user. A non-human User **50** includes any device, system, program, and/or other mechanism for operating or controlling Application Program **18**, Avatar **605**, Computing Device **70**, and/or elements thereof. For example, User **50** may issue an operating direction to Application Program **18** responsive to which Application Program's **18** instructions or instruction sets may be executed by Processor **11** to perform a desired operation with/on Avatar **605**. User's **50** operating directions comprise any user inputted data (i.e. values, text, symbols, etc.), directions (i.e. move right, move up, move forward, copy an item, click on a link, etc.), instructions or instruction sets (i.e. manually inputted instructions or instruction sets, etc.), and/or other inputs or information. A non-human User **50** can utilize more suitable interfaces instead of, or in addition to, Human-machine Interface **23** and/or Display **21** for controlling Application Program **18**, Avatar **605**, Computing Device **70**, and/or elements thereof. Examples of such interfaces include an application programming interface (API), bridge (i.e. bridge between applications, devices, or systems, etc.), driver, socket, direct or operative connection, handle, function/routine/subroutine, and/or other interfaces.

Avatar **605** may be or comprises an object of Application Program **18**. While Avatar **605** may include any features, functionalities, and embodiments of Object **615** (later described), Avatar **605** is distinguished herein to portray the relationships and/or interactions between Avatar **605** and other Objects **615** of Application Program **18**. In some aspects, Avatar **605** includes a User **50**-controllable object of Application Program **18**. Avatar **605** may, therefore, be a representation of User **50** or of User's **50** actions, thoughts, and/or other expressions. In some designs, Avatar **605** includes a 2D model, a 3D model, a 2D shape (i.e. point, line, square, rectangle, circle, triangle, etc.), a 3D shape (i.e. cube, sphere, etc.), a graphical user interface (GUI) element, a picture, and/or other models, shapes, elements, or objects. Avatar **605** may perform one or more operations within Application Program **18**. For example, Avatar **605** may perform operations including moving, maneuvering, jumping, running, shooting, and/or other operations within a game or virtual world Application Program **18**. While all possible variations of operations on/by/with Avatar **605** are too voluminous to list and limited only by Avatar's **605** and/or Application Program's **18** design, and/or User's **50** utilization, other operations on/by/with Avatar **605** are within the scope of this disclosure.

Object Processing Unit **140** comprises the functionality for obtaining information of interest on objects of Application Program **18**, and/or other functionalities. As such, Object Processing Unit **140** can be used to obtain objects and/or their properties in Avatar's **605** surrounding within Application Program **18**. Avatar's **605** surrounding may include or be defined by Area of Interest **450** (later described), part of Application Program **18** that is shown to User **50** (i.e. on a display, via a graphical user interface, etc.), the entire Application Program **18**, any part of Application Program **18**, and/or other techniques. In some embodiments, Object Processing Unit **140** comprises the functionality for creating or generating Collection of Object Representations **525** (also referred to as Coll of Obj Rep or other suitable name or reference) and storing one or more Object Representations **625** (also referred to simply as object representations, representations of objects, or other suitable name or reference), Object Properties **630** (also referred to simply as object properties or other suitable name or reference), and/or other elements or information into the Collection of Object Representations **525**. As such, Collection of Object Representations **525** comprises the functionality for storing one or

more Object Representations **625**, Object Properties **630**, and/or other elements or information. In some designs, Object Representation **625** may include a representation of an object (i.e. Object **615** [later described], etc.) in Avatar's **605** surrounding within Application Program **18**. As such, Object Representation **625** may include any information related to an object. In other designs, Object Representation **625** may include or be replaced with an object itself, in which case Object Representation **625** as an element can be omitted. In some aspects, Collection of Object Representations **525** includes one or more Object Representations **625**, Object Properties **630**, and/or other elements or information related to objects in Avatar's **605** surrounding at a particular time. Collection of Object Representations **525** may, therefore, include knowledge (i.e. unit of knowledge, etc.) of Avatar's **605** circumstance including objects with various properties at a particular time. In some designs, a Collection of Object Representations **525** may include or be associated with a time stamp (not shown), order (not shown), or other time related information. For example, one Collection of Object Representations **525** may be associated with time stamp t**1**, another Collection of Object Representations **525** may be associated with time stamp t**2**, and so on. Time stamps t**1**, t**2**, etc. may indicate the times of generating Collections of Object Representations **525**, for instance. In other embodiments, Object Processing Unit **140** comprises the functionality for creating or generating a stream of Collections of Object Representations **525**. A stream of Collections of Object Representations **525** may include one Collection of Object Representations **525** or a group, sequence, or other plurality of Collections of Object Representations **525**. In some aspects, a stream of Collections of Object Representations **525** includes one or more Collections of Object Representations **525**, and/or other elements or information related to objects in Avatar's **605** surrounding over time. A stream of Collections of Object Representations **525** may, therefore, include knowledge (i.e. unit of knowledge, etc.) of Avatar's **605** circumstance including objects with various properties over time. As circumstances including objects with various properties in Avatar's **605** surrounding change (i.e. objects and/or their properties change, move, act, transform, etc.) over time, this change may be captured in a stream of Collections of Object Representations **525**. In some designs, each Collection of Object Representations **525** in a stream may include or be associated with the aforementioned time stamp, order, or other time related information. For example, one Collection of Object Representations **525** in a stream may be associated with order **1**, a next Collection of Object Representations **525** in the stream may be associated with order **2**, and so on. Orders **1**, **2**, etc. may indicate the orders or places of Collections of Object Representations **525** within a stream (i.e. sequence, etc.), for instance. In some implementations, Object Processing Unit **140** and/or any of its elements or functionalities can be included or embedded in Computing Device **70**, Processor **11**, Application Program **18**, and/or other elements. Object Processing Unit **140** can be provided in any suitable configuration.

Examples of Objects **615** (also referred to simply as objects, etc.) include models of a person, animal, tree, rock, building, vehicle, and/or others in a context of a computer game, virtual world, 3D or 2D graphics Application Program **18**, and/or others. More generally, examples of Objects **615** include a 2D model, a 3D model, a 2D shape (i.e. point, line, square, rectangle, circle, triangle, etc.), a 3D shape (i.e. cube, sphere, etc.), a graphical user interface (GUI) element, a form element (i.e. text field, radio button, push button, check box, etc.), a data or database element, a spreadsheet element, a link, a picture, a text (i.e. character, word, etc.), a number, and/or others in a context of a web browser, a media application, a word processing application, a spreadsheet application, a database application, a forms-based application, an operating system, a device/system control application, and/or others. Object **615** may perform operations within Application Program **18**. In one example, a person Object **615** may perform operations including moving, maneuvering, jumping, running, shooting, and/or other operations within a computer game, virtual world, and/or 3D or 2D graphics Application Program **18**. In another example, a character Object **615** may perform operations including appearing (i.e. when typed, etc.), disappearing (i.e. when deleted, etc.), formatting (i.e. bolding, italicizing, underlining, coloring, resizing, etc.), and/or other operations within a word processing Application Program **18**. In a further example, a picture Object **615** may perform operations including resizing, repositioning, rotating, deforming, and/or other operations within a graphics Application Program **18**. While all possible variations of operations on/by/with Object **615** are too voluminous to list and limited only by Object's **615** and/or Application Program's **18** design, and/or User's **50** utilization, other operations on/by/with Object **615** are within the scope of this disclosure. In some aspects, any part of Object **615** can be identified as an Object **615** itself. For instance, instead of or in addition to identifying a building as an Object **615**, a window, door, roof, and/or other parts of the building can be identified as Objects **615**. In general, Object **615** may include any object or part thereof that can be obtained or recognized.

Examples of Object Properties **630** (i.e. also referred to simply as object properties, etc.) include existence of Object **615**, type of Object **615** (i.e. person, animal, tree, rock, building, vehicle, etc.), identity of Object **615** (i.e. name, identifier, etc.), distance of Object **615**, bearing/angle of Object **615**, location of Object **615** (i.e. distance and bearing/angle from a known point, coordinates, etc.), shape/size of Object **615** (i.e. scale, height, width, depth, computer model, etc.), activity of Object **615** (i.e. motion, gestures, etc.), and/or other properties of Object **615**. Type of Object **615**, for example, may include any classification of objects ranging from detailed such as person, animal, tree, rock, building, vehicle, etc. to generalized such as biological object, nature object, manmade object, etc., or models thereof, including their sub-types. Location of Object **615**, for example, can include a relative location such as one defined by distance and bearing/angle from a known point or location (i.e. Avatar **605** location, etc.). Location of Object **615**, for example, can also include absolute location such as one defined by object coordinates. In general, Object Property **630** may include any attribute of Object **615** (i.e. existence of Object **615**, type of Object **615**, identity of Object **615**, shape/size of Object **615**, etc.), any relationship of Object **615** with Avatar **605**, other Object **615**, or the environment (i.e. distance of Object **615**, bearing/angle of Object **615**, friend/foe relationship, etc.), and/or other information related to Object **615**.

In some embodiments, Object Processing Unit **140** can be utilized for obtaining properties of Objects **615** in Avatar's **605** surrounding within Application Program **18**. In some designs, an engine, environment, or other system used to implement Application Program **18** includes functions for providing properties or other information on Objects **615**. Object Processing Unit **140** can obtain Object Properties **630** by utilizing the functions. In some aspects, existence of Object **615** in a 2D or 3D engine or environment can be

obtained by utilizing functions such as GameObject.FindObjectsOfType(GameObject), GameObject.FindGameObjectsWithTag("TagN"), or GameObject.Find("ObjectN") in Unity 3D Engine; GetAllActorsOfClass( ) or IsActorInitialized( ) in Unreal Engine; and/or other functions, procedures, or methods in other 2D or 3D engines or environments. In other aspects, type or other classification (i.e. person, animal, tree, rock, building, vehicle, etc.) of Object **615** in a 2D or 3D engine or environment can be obtained by utilizing functions such as GetClassName(ObjectN) or ObjectN.getType( ) in Unity 3D Engine; ActorN.GetClass( ) in Unreal Engine; ObjectN.getClassName( ) or ObjectN.getType( ) in Torque 3D Engine; and/or other functions, procedures, or methods in other 2D or 3D engines or environments. In further aspects, identity of Object **615** in a 2D or 3D engine or environment can be obtained by utilizing functions such as ObjectN.name or ObjectN.GetInstanceID( ) in Unity 3D Engine; ActorN.GetObjectName( ) or ActorN.GetUniqueID( ) in Unreal Engine; ObjectN.getName( ) or ObjectN.getID( ) in Torque 3D Engine; and/or other functions, procedures, or methods in other 2D or 3D engines or environments. In further aspects, distance of Object **615** relative to Avatar **605** in a 2D or 3D engine or environment can be obtained by utilizing functions such as VectorN.Distance(ObjectA.transform.position, ObjectB.transform.position) in Unity 3D Engine; GetDistanceTo(ActorA, ActorB) in Unreal Engine; VectorDist(VectorA, VectorB) or VectorDist(ObjectA.getPositiono,ObjectB.getPosition( ) in Torque 3D Engine; and/or other functions, procedures, or methods in other 2D or 3D engines or environments. In further aspects, angle, bearing, or direction of Object **615** relative to Avatar **605** in a 2D or 3D engine or environment can be obtained by utilizing functions such as ObjectB.transform.position—ObjectA.transform.position in Unity 3D Engine; FindLookAtRotation(TargetVector, StartVector) or ActorB→GetActorLocation( )—ActorA→GetActorLocation( ) in Unreal Engine; ObjectB→getPosition( )-ObjectA→getPosition( ) in Torque 3D Engine; and/or other functions, procedures, or methods in other 2D or 3D engines or environments. In further aspects, location of Object **615** in a 2D or 3D engine or environment can be obtained by utilizing functions such as ObjectN.transform.position in Unity 3D Engine; ActorN.GetActorLocation( ) in Unreal Engine; ObjectN.getPosition( ) in Torque 3D Engine; and/or other similar functions, procedures, or methods in other 2D or 3D engines or environments. In another example, location (i.e. coordinates, etc.) of Object **615** on a screen can be obtained by utilizing WorldToScreen( ) or other similar function or method in various 2D or 3D engines or environments. In some designs, distance, angle/bearing, and/or other properties of Object **615** relative to Avatar **605** can then be calculated, inferred, derived, or estimated from Object's **615** and Avatar's **605** location information. Object Processing Unit **140** may include computational functionalities to perform such calculations, inferences, derivations, or estimations by utilizing, for example, geometry, trigonometry, Pythagorean theorem, and/or other theorems, formulas, or disciplines. In further aspects, shape/size of Object **615** in a 2D or 3D engine or environment can be obtained by utilizing functions such as Bounds.size, ObjectN.transform.localScale, or ObjectN.transform.lossyScale in Unity 3D Engine; ActorN.GetActorBounds( ), ActorN.GetActorScale( ), or ActorN.GetActorScale3D( ) in Unreal Engine; ObjectN.getObjectBox( ) or ObjectN.getScale( ) in Torque 3D Engine; and/or other similar functions, procedures, or methods in other 2D or 3D engines or environments. In some designs, detailed shape of Object **615** can be obtained by accessing the object's mesh or computer model. In general, any of the aforementioned and/or other properties of Object **615** can be obtained by accessing a scene graph or other data structure used for organizing objects in a particular engine or environment, finding a specific Object **615**, and obtaining or reading any property from the Object **615**. Such accessing can be performed by using the engine's or environment's functions for accessing objects in the scene graph or other data structure or by directly accessing the scene graph or other data structure. In some designs, functions and/or other instructions for obtaining properties or other information on Objects **615** of Application Program **18** can be inserted or utilized in Application Program's **18** source code. In other designs, functions and/or other instructions for obtaining properties or other information on Objects **615** of Application Program **18** can be inserted into Application Program **18** through manual, automatic, dynamic, or just-in-time (JIT) instrumentation (later described). In further designs, functions and/or other instructions for providing properties or other information on Objects **615** of Application Program **18** can be inserted into Application Program **18** through utilizing dynamic code, dynamic class loading, reflection, and/or other functionalities of a programming language or platform; utilizing dynamic, interpreted, and/or scripting programming languages; utilizing metaprogramming; and/or utilizing other techniques (later described). Object Processing Unit **140** may include any features, functionalities, and embodiments of Acquisition Interface **120**, Modification Interface **130**, and/or other elements. One of ordinary skill in art will understand that the aforementioned techniques for obtaining objects and/or their properties are described merely as examples of a variety of possible implementations, and that while all possible techniques for obtaining objects and/or their properties are too voluminous to describe, other techniques for obtaining objects and/or their properties known in art are within the scope of this disclosure. It should be noted that Unity 3D Engine, Unreal Engine, and Torque 3D Engine are used merely as examples of a variety of engines, environments, or systems that can be used to implement Application Program **18** and any of the aforementioned functionalities may be provided in other engines, environments, or systems. Also, in some embodiments, Application Program **18** may not use any engine, environment, or system for its implementation, in which case the aforementioned functionalities can be implemented within Application Program **18**. In general, the disclosed devices, systems, and methods are independent of the engine, environment, or system used to implement Application Program **18**.

In some embodiments of Application Programs **18** that do not comprise Avatar **605** or rely on Avatar **605** for their operation, Object Processing Unit **140** may obtain objects and/or their properties in Application Program **18** or a part thereof. For example, Object Processing Unit **140** can obtain objects and/or their properties in the entire Application Program **18**, a part of Application Program **18** that is shown to User **50** (i.e. on a display, via a graphical user interface, etc.), or any part or area of interest (later described) of Application Program **18**. In such embodiments, Object Processing Unit **140** can create or generate Collections of Object Representations **525** or streams of Collections of Object Representations **525** comprising knowledge (i.e. unit of knowledge, etc.) of Application Program's **18** circumstances including objects with various properties. It should be noted that a reference to Avatar **605** may include or be substituted with a reference to Application Program **18** and/or other processing element, and vice versa, depending

on context (i.e. whether Avatar's **605** or Application Program's **18** operation is being learned and/or used, etc.). Also, a reference to operating and/or autonomous operating of Avatar **605** may include or be substituted with a reference to operating and/or autonomous operating of Application Program **18** and/or other processing element depending on context.

Referring to FIG. **3**, an embodiment of utilizing Picture Renderer **91** and Picture Recognizer **92** is illustrated.

Picture Renderer **91** comprises the functionality for rendering or generating one or more digital pictures, and/or other functionalities. Picture Renderer **91** comprises the functionality for rendering or generating one or more digital pictures of Application Program **18**. In some aspects, as a camera is used to capture pictures of a physical environment, Picture Renderer **91** can be used to render or generate pictures of a computer modeled or represented environment. As such, Picture Renderer **91** can be used to render or generate views of Application Program **18**. In some designs, Picture Renderer **91** can be used to render or generate one or more digital pictures depicting a view of an Avatar's **605** visual surrounding in a 3D Application Program **18** (i.e. 3D computer game, virtual world application, CAD application, etc.). In one example, a view may include a first-person view or perspective such as a view through an avatar's eyes that shows objects around the avatar, but does not typically show the avatar itself. First-person view may sometimes include the avatar's hands, feet, other body parts, and/or objects that the avatar is holding. In another example, a view may include a third-person view or perspective such as a view that shows an avatar as well as objects around the avatar from an observer's point of view. In a further example, a view may include a view from a front of an avatar. In a further example, a view may include a view from a side of an avatar. In a further example, a view may include any stationary or movable view such as a view through a simulated camera in a 3D Application Program **18**. In other designs, Picture Renderer **91** can be used to render or generate one or more digital pictures depicting a view of a 2D Application Program **18**. In one example, a view may include a screenshot or portion thereof of a 2D Application Program **18**. In a further example, a view may include an area of interest of a 2D Application Program **18**. In a further example, a view may include a top-down view of a 2D Application Program **18**. In a further example, a view may include a side-on view of a 2D Application Program **18**. Any other view can be utilized in alternate designs. Any view utilized in a 3D Application Program **18** can similarly be utilized in a 2D Application Program **18** as applicable, and vice versa. In some implementations, Picture Renderer **91** may include any graphics processing device, apparatus, system, or application that can render or generate one or more digital pictures from a computer (i.e. 3D, 2D, etc.) model or representation. In some aspects, rendering, when used casually, may refer to rendering or generating one or more digital pictures from a computer model or representation, providing the one or more digital pictures to a display device, and/or displaying of the one or more digital pictures on a display device. In some embodiments, Picture Renderer **91** can be a program executing or operating on Processor **11**. In one example, Picture Renderer **91** can be provided in a rendering engine such as Direct3D, OpenGL, Mantle, and/or other programs or systems for rendering or processing 3D or 2D graphics. In other embodiments, Picture Renderer **91** can be part of, embedded into, or built into Processor **11**. In further embodiments, Picture Renderer **91** can be a hardware element coupled to Processor **11** and/or other elements. In further embodiments, Picture Renderer **91** can be a program or hardware element that is part of or embedded into another element. In one example, a graphics card and/or its graphics processing unit (i.e. GPU, etc.) may typically include Picture Renderer **91**. In another example, ACAAO Unit **100** may include Picture Renderer **91**. In a further example, Application Program **18** may include Picture Renderer **91**. In general, Picture Renderer **91** can be implemented in any suitable configuration to provide its functionalities. Picture Renderer **91** may render or generate one or more digital pictures or streams of digital pictures (i.e. motion pictures, video, etc.) in various formats examples of which include JPEG, GIF, TIFF, PNG, PDF, MPEG, AVI, FLV, MOV, RM, SWF, WMV, DivX, and/or others. In some designs, Picture Renderer **91** can render or generate different digital pictures of Avatar's **605** visual surrounding or of views of Application Program **18** for displaying on Display **21** and for facilitating object recognition functionalities herein. For example, a third-person view may be displayed on Display **21** for User **50** to see and a first-person view may be used to facilitate object recognition functionalities herein. In some implementations of non-graphical Application Programs **18** such as simulations, calculations, and/or others, Picture Renderer **91** may render or generate one or more digital pictures of Avatar's **605** visual surrounding or of views of Application Program **18** to facilitate object recognition functionalities herein where the one or more digital pictures are never displayed. In some aspects, instead of or in addition to Picture Renderer **91**, one or more digital pictures of Avatar's **605** visual surrounding or of views of Application Program **18** can be obtained from any element of a computing device or system that can provide such digital pictures. Examples of such elements include a graphics circuit, a graphics system, a graphics driver, a graphics interface, and/or others.

Picture Recognizer **92** comprises the functionality for detecting or recognizing objects and/or their properties in visual data, and/or other disclosed functionalities. Visual data includes digital motion pictures, digital still pictures, and/or other visual data. Examples of file formats that can be utilized to store visual data include JPEG, GIF, TIFF, PNG, PDF, MPEG, AVI, FLV, MOV, RM, SWF, WMV, DivX, and/or other file formats. In some designs, Picture Recognizer **92** can be used for detecting or recognizing objects and/or their properties in one or more digital pictures from Picture Renderer **91**. For example, Picture Recognizer **92** can be utilized in detecting or recognizing existence of an object, type of an object, identity of an object, shape/size of an object, activity of an object, and/or other properties of an object. In general, Picture Recognizer **92** can be used for any operation supported by Picture Recognizer **92**. Picture Recognizer **92** may detect or recognize an object and/or its properties as well as track the object and/or its properties in one or more digital pictures or streams of digital pictures (i.e. motion pictures, video, etc.). In the case of a person, Picture Recognizer **92** may detect or recognize a human head or face, upper body, full body, or portions/combinations thereof. In some aspects, Picture Recognizer **92** may detect or recognize objects and/or their properties from a digital picture by comparing regions of pixels from the digital picture with collections of pixels comprising known objects and/or their properties. The collections of pixels comprising known objects and/or their properties can be learned or manually, programmatically, or otherwise defined. The collections of pixels comprising known objects and/or their properties can be stored in any data structure or repository (i.e. one or more files, database, etc.) that resides

locally on Computing Device **70**, or remotely on a remote computing device (i.e. server, cloud, etc.) accessible over a network or an interface. In other aspects, Picture Recognizer **92** may detect or recognize objects and/or their properties from a digital picture by comparing features (i.e. lines, edges, ridges, corners, blobs, regions, etc.) of the digital picture with features of known objects and/or their properties. The features of known objects and/or their properties can be learned or manually, programmatically, or otherwise defined. The features of known objects and/or their properties can be stored in any data structure or repository (i.e. neural network, one or more files, database, etc.) that resides locally on Computing Device **70**, or remotely on a remote computing device (i.e. server, cloud, etc.) accessible over a network or an interface. Typical steps or elements in a feature oriented picture recognition include pre-processing, feature extraction, detection/segmentation, decision-making, and/or others, or a combination thereof, each of which may include its own sub-steps or sub-elements depending on the application. In further aspects, Picture Recognizer **92** may detect or recognize multiple objects and/or their properties from a digital picture using the aforementioned pixel or feature comparisons, and/or other detection or recognition techniques. For example, a picture may depict two objects in two of its regions both of which Picture Recognizer **92** can detect simultaneously. In further aspects, where objects and/or their properties span multiple pictures, Picture Recognizer **92** may detect or recognize objects and/or their properties by applying the aforementioned pixel or feature comparisons and/or other detection or recognition techniques over a stream of digital pictures (i.e. motion picture, video, etc.). For example, once an object is detected in a digital picture (i.e. frame, etc.) of a stream of digital pictures (i.e. motion picture, video, etc.), the region of pixels comprising the detected object or the object's features can be searched in other pictures of the stream of digital pictures, thereby tracking the object through the stream of digital pictures. In further aspects, Picture Recognizer **92** may detect or recognize an object's activities by identifying and/or analyzing differences between a detected region of pixels of one picture (i.e. frame, etc.) and detected regions of pixels of other pictures in a stream of digital pictures. For example, a region of pixels comprising a person's face can be detected in multiple consecutive pictures of a stream of digital pictures (i.e. motion picture, video, etc.). Differences among the detected regions of the consecutive pictures may be identified in the mouth part of the person's face to indicate smiling or speaking activity. In further aspects, Picture Recognizer **92** may detect or recognize objects and/or their properties using one or more artificial neural networks, which may include statistical techniques. Examples of artificial neural networks that can be used in Picture Recognizer **92** include convolutional neural networks (CNNs), time delay neural networks (TDNNs), deep neural networks, and/or others. In one example, picture recognition techniques and/or tools involving convolutional neural networks may include identifying and/or analyzing tiled and/or overlapping regions or features of a digital picture, which may then be used to search for pictures with matching regions or features. In another example, features of different convolutional neural networks responsible for spatial and temporal streams can be fused to detect objects and/or their properties in streams of digital pictures (i.e. motion pictures, videos, etc.). In general, Picture Recognizer **92** may include any machine learning, deep learning, and/or other artificial intelligence techniques. Any other techniques known in art can be utilized in Picture Recognizer **92**. For example, thresholds for similarity, statistical techniques, and/or optimization techniques can be utilized to determine a match in any of the above-described detection or recognition techniques.

Various aspects or properties of digital pictures or pixels can be taken into account by Picture Recognizer **92** in any of the recognizing or comparisons. Examples of such aspects or properties include color adjustment, size adjustment, content manipulation, transparency (i.e. alpha channel, etc.), use of mask, and/or others. In some implementations, as digital pictures can be captured or generated by various equipment, in various environments, and under various lighting conditions, Picture Recognizer **92** can adjust lighting or color of pixels or otherwise manipulate pixels before or during comparison. Lighting or color adjustment (also referred to as gray balance, neutral balance, white balance, etc.) may generally include manipulating or rebalancing the intensities of the colors (i.e. red, green, and/or blue if RGB color model is used, etc.) of one or more pixels. For example, Picture Recognizer **92** can adjust lighting or color of some or all pixels of one picture to make it more comparable to another picture. Picture Recognizer **92** can also incrementally adjust the pixels such as increasing or decreasing the red, green, and/or blue pixel values by a certain amount in each cycle of comparisons in order to find a similarity or match at one of the incremental adjustment levels. Any of the publically available, custom, or other lighting or color adjustment techniques or programs can be utilized such as color filters, color balancing, color correction, and/or others. In other implementations, Picture Recognizer **92** can resize or otherwise transform a digital picture before or during comparison. Such resizing or transformation may include increasing or decreasing the number of pixels of a digital picture. For example, Picture Recognizer **92** can increase or decrease the size of a digital picture proportionally (i.e. increase or decrease length and/or width keeping aspect ratio constant, etc.) to equate its size with the size of another digital picture. Picture Recognizer **92** can also incrementally resize a digital picture such as increasing or decreasing the size of the digital picture proportionally by a certain amount in each cycle of comparisons in order to find a similarity or match at one of the incremental sizes. Any of the publically available, custom, or other digital picture resizing techniques or programs can be utilized such as nearest-neighbor interpolation, bilinear interpolation, bicubic interpolation, and/or others. In further implementations, Picture Recognizer **92** can manipulate content (i.e. all pixels, one or more regions, one or more depicted objects, etc.) of a digital picture before or during comparison. Such content manipulation may include moving, centering, aligning, resizing, transforming, and/or otherwise manipulating content of a digital picture. For example, Picture Recognizer **92** can move, center, or align content of one picture to make it more comparable to another picture. Any of the publically available, custom, or other digital picture manipulation techniques or programs can be utilized such as pixel moving, warping, distorting, aforementioned interpolations, and/or others. In further implementations, in digital pictures comprising transparency features or functionalities, Picture Recognizer **92** can utilize a threshold for acceptable number or percentage transparency difference. Alternatively, transparency can be applied to one or more pixels of a digital picture and color difference may then be determined between compared pixels taking into account the transparency related color effect. Alternatively, transparent pixels can be excluded from comparison. In further implementations, certain regions or subsets of pixels can be ignored or excluded

during comparison using a mask. In general, any region or subset of a picture determined to contain no content of interest can be excluded from comparison using a mask. Examples of such regions or subsets include background, transparent or partially transparent regions, regions comprising insignificant content, or any arbitrary region or subset. Picture Recognizer **92** can perform any other preprocessing or manipulation of digital pictures or pixels before or during recognizing or comparison.

In some exemplary embodiments, object recognition techniques and/or tools such as OpenCV (Open Source Computer Vision) library, CamFind API, Kooaba, 6 px API, Dextro API, and/or others can be utilized for detecting or recognizing objects and/or their properties in digital pictures. In some aspects, picture recognition techniques and/or tools involve identifying and/or analyzing features such as lines, edges, ridges, corners, blobs, regions, and/or their relative positions, sizes, shapes, etc., which may then be used to search for pictures with matching features. For example, OpenCV library can detect an object (i.e. person, animal, vehicle, rock, etc.) and/or its properties in one or more digital pictures from Picture Renderer **91** or stored in an electronic repository, which can then be utilized in ACAAO Unit **100**, Artificial Intelligence Unit **110**, and/or other elements. In other exemplary embodiments, facial recognition techniques and/or tools such as OpenCV (Open Source Computer Vision) library, Animetrics FaceR API, Lambda Labs Facial Recognition API, Face++ SDK, Neven Vision (also known as N-Vision) Engine, and/or others can be utilized for detecting or recognizing faces in digital pictures. In some aspects, facial recognition techniques and/or tools involve identifying and/or analyzing facial features such as the relative position, size, and/or shape of the eyes, nose, cheekbones, jaw, etc., which may then be used to search for pictures with matching features. For example, FaceR API can detect a person's face in one or more digital pictures from Picture Renderer **91** or stored in an electronic repository, which can then be utilized in ACAAO Unit **100**, Artificial Intelligence Unit **110**, and/or other elements.

It should be noted that Picture Renderer **91** and Picture Recognizer **92** can optionally be used to detect objects and/or their properties that cannot not be obtained from Application Program **18** or from an engine, environment, or system used to implement Application Program **18**. Picture Renderer **91** and Picture Recognizer **92** can also optionally be used where Picture Renderer **91** and Picture Recognizer **92** offer superior performance in detecting objects and/or their properties. Picture Renderer **91** and Picture Recognizer **92** can also optionally be used to confirm objects and/or their properties obtained or detected by other means. For example, identity of an object, type of an object, and/or action of an object, if needed, can be recognized or confirmed through picture processing of Picture Renderer **91** and Picture Recognizer **92**. Picture Renderer **91** and Picture Recognizer **92** can be omitted depending on implementation.

Referring to FIG. **4**, an embodiment of utilizing Sound Renderer **96** and Sound Recognizer **97** is illustrated.

Sound Renderer **96** comprises the functionality for rendering or generating digital sound, and/or other functionalities. Sound Renderer **96** comprises the functionality for rendering or generating digital sound of Application Program **18**. In some aspects, as a microphone is used to capture sound of a physical environment, Sound Renderer **96** can be used to render or generate sound of a computer modeled or represented environment. As such, Sound Renderer **96** can be used to render or generate sound of Application Program **18**. In some designs, Sound Renderer **96** can be used to render or generate digital sound from Avatar's **605** surrounding in a 3D Application Program **18** (i.e. 3D computer game, virtual world application, CAD application, etc.). For example, emission of a sound from a sound source may be simulated/modeled in a virtual space of a 3D Application Program **18**, propagation of the sound may be simulated/modeled through the virtual space including any scattering, reflections, refractions, diffractions, and/or other effects, and the sound may be rendered or generated as perceived by a listener (i.e. Avatar **605**, etc.). In other designs, Sound Renderer **96** can be used to render or generate digital sound of a 2D Application Program **18** which may include any of the aforementioned and/or other sound simulation/modeling as applicable to 2D spaces. In further designs, Sound Renderer **96** can be optionally omitted in a simple Application Program **18** where no sound simulation/modeling is needed or where sounds may simply be played. In some implementations, Sound Renderer **96** may include any sound processing device, apparatus, system, or application that can render or generate digital sound. In some aspects, rendering, when used casually, may refer to rendering or generating digital sound from a computer model or representation, providing digital sound to a speaker or headphones, and/or producing the sound by a speaker or headphones. In some embodiments, Sound Renderer **96** can be a program executing or operating on Processor **11**. In one example, Sound Renderer **96** can be provided in a rendering engine such as SoundScape Renderer, SLAB Spatial Audio Renderer, Uni-Verse Sound Renderer, Crepo Sound Renderer, and/or other programs or systems for rendering or processing sound. In another example, various engines or environments such as Unity 3D Engine, Unreal Engine, Torque 3D Engine, and/or others provide built-in sound renderers. In other embodiments, Sound Renderer **96** can be part of, embedded into, or built into Processor **11**. In further embodiments, Sound Renderer **96** can be a hardware element coupled to Processor **11** and/or other elements. In further embodiments, Sound Renderer **96** can be a program or hardware element that is part of or embedded into another element. In one example, a sound card and/or its processing unit may include Sound Renderer **96**. In another example, ACAAO Unit **100** may include Sound Renderer **96**. In a further example, Application Program **18** may include Sound Renderer **96**. In general, Sound Renderer **96** can be implemented in any suitable configuration to provide its functionalities. Sound Renderer **96** may render or generate digital sound in various formats examples of which include WAV, WMA, AIFF, MP3, RA, OGG, and/or others. In some designs, Sound Renderer **96** can render or generate different digital sound of an Application Program **18** for production on a speaker or headphones and for facilitating object recognition functionalities herein. For example, sound of Avatar's **605** shooting may be produced by a speaker or headphones for User **50** to hear and sound of various objects in Avatar's **605** surrounding may be used to facilitate object recognition functionalities herein. In some implementations of non-acoustic Application Programs **18** such as simulations, calculations, and/or others, Sound Renderer **96** may render or generate digital sound as perceived by Avatar **605** to facilitate object recognition functionalities herein where the sound is never produced on a speaker or headphones. In some aspects, instead of or in addition to Sound Renderer **96**, digital sound perceived by Avatar **605** can be obtained from any element of a computing device or system that can provide such digital sound.

Examples of such elements include an audio circuit, an audio system, an audio driver, an audio interface, and/or others.

Sound Recognizer **97** comprises the functionality for detecting or recognizing objects and/or their properties in audio data, and/or other disclosed functionalities. Audio data includes digital sound, and/or other audio data. Examples of file formats that can be utilized to store audio data include WAV, WMA, AIFF, MP3, RA, OGG, and/or other file formats. In some designs, Sound Recognizer **97** can be used for detecting or recognizing objects and/or their properties in digital sound from Sound Renderer **96**. In the case of a person, Sound Recognizer **97** may detect or recognize human voice. For example, Sound Recognizer **97** can be utilized in detecting or recognizing existence of an object, type of an object, identity of an object, activity of an object, and/or other properties of an object. In general, Sound Recognizer **97** can be used for any operation supported by Sound Recognizer **97**. In some aspects, Sound Recognizer **97** may detect or recognize an object and/or its properties from a digital sound by comparing collections of sound samples from the digital sound with collections of sound samples of known objects and/or their properties. The collections of sound samples of known objects and/or their properties can be learned, or manually, programmatically, or otherwise defined. The collections of sound samples of known objects and/or their properties can be stored in any data structure or repository (i.e. one or more files, database, etc.) that resides locally on Computing Device **70**, or remotely on a remote computing device (i.e. server, cloud, etc.) accessible over a network or an interface. In other aspects, Sound Recognizer **97** may detect or recognize an object and/or its properties from a digital sound by comparing features from the digital sound with features of sounds of known objects and/or their properties. The features of sounds of known objects and/or their properties can be learned, or manually, programmatically, or otherwise defined. The features of sounds of known objects and/or their properties can be stored in any data structure or repository (i.e. neural network, one or more files, database, etc.) that resides locally on Computing Device **70**, or remotely on a remote computing device (i.e. server, cloud, etc.) accessible over a network or an interface. Typical steps or elements in a feature oriented sound recognition include pre-processing, feature extraction, acoustic modeling, language modeling, and/or others, or a combination thereof, each of which may include its own sub-steps or sub-elements depending on the application. In further aspects, Sound Recognizer **97** may detect or recognize a variety of sounds from digital sound using the aforementioned sound sample or feature comparisons, and/or other detection or recognition techniques. For example, sound of a person, animal, vehicle, and/or other sounds can be detected by Sound Recognizer **97**. In further aspects, Sound Recognizer **97** may detect or recognize sounds using Hidden Markov Models (HMM), Artificial Neural Networks, Dynamic Time Warping (DTW), Gaussian Mixture Models (GMM), and/or other models or techniques, or a combination thereof. Some or all of these models or techniques may include statistical techniques. Examples of artificial neural networks that can be used in Sound Recognizer **97** include recurrent neural networks, time delay neural networks (TDNNs), deep neural networks, convolutional neural networks, and/or others. In general, Sound Recognizer **97** may include any machine learning, deep learning, and/or other artificial intelligence techniques. Any other techniques known in art can be utilized in Sound Recognizer **97**. For example, thresholds for similarity, statistical techniques, and/or optimization techniques can be utilized to determine a match in any of the above-described detection or recognition techniques.

In some exemplary embodiments, operating system's sound recognition functionalities such as iOS's Voice Services, Siri, and/or others can be utilized in Sound Recognizer **97**. For example, iOS Voice Services can detect an object (i.e. person, etc.) and/or its properties in digital sound from Sound Renderer **96** or stored in an electronic repository, which can then be utilized in ACAAO Unit **100**, Artificial Intelligence Unit **110**, and/or other elements. In other exemplary embodiments, Java Speech API (JSAPI) implementation such as The Cloud Garden, Sphinx, and/or others can be utilized in Sound Recognizer **97**. For example, Cloud Garden JSAPI can detect an object (i.e. person, animal, vehicle, etc.) and/or its properties in digital sound from Sound Renderer **96** or stored in an electronic repository, which can then be utilized in ACAAO Unit **100**, Artificial Intelligence Unit **110**, and/or other elements. Any other programming language's or platform's speech or sound processing API can similarly be utilized. In further exemplary embodiments, applications or engines providing Sound recognition functionalities such as HTK (Hidden Markov Model Toolkit), Kaldi, OpenEars, Dragon Mobile, Julius, iSpeech, CeedVocal, and/or others can be utilized in Sound Recognizer **97**. For example, Kaldi SDK can detect an object (i.e. person, animal, vehicle, etc.) and/or its properties in digital sound from Sound Renderer **96** or stored in an electronic repository, which can then be utilized in ACAAO Unit **100**, Artificial Intelligence Unit **110**, and/or other elements.

It should be noted that Sound Renderer **96** and Sound Recognizer **97** can optionally be used to detect objects and/or their properties that cannot not be obtained from Application Program **18** or from an engine, environment, or system used to implement Application Program **18**. Sound Renderer **96** and Sound Recognizer **97** can also optionally be used where Sound Renderer **96** and Sound Recognizer **97** offer superior performance in detecting objects and/or their properties. Sound Renderer **96** and Sound Recognizer **97** can also optionally be used to confirm objects and/or their properties obtained or detected by other means. For example, identity of an object, type of an object, and/or activity of an object, if needed, can be recognized or confirmed through sound processing of Sound Renderer **96** and Sound Recognizer **97**. Sound Renderer **96** and Sound Recognizer **97** can be omitted depending on implementation.

One of ordinary skill in art will understand that the aforementioned techniques for detecting or recognizing objects and/or their properties using pictures and sounds are described merely as examples of a variety of possible implementations, and that while all possible techniques for detecting or recognizing objects and/or their properties are too voluminous to describe, other techniques for detecting or recognizing objects and/or their properties known in art are within the scope of this disclosure. Also, any signal processing technique known in art that can facilitate the disclosed functionalities can be utilized in various embodiments. Any combination of the aforementioned and/or other renderers, object detecting or recognizing techniques, signal processing techniques, and/or other elements or techniques can be used in various embodiments.

Referring to FIGS. **5A-5B**, an exemplary embodiment of Objects **615** (also referred to simply as objects or other

suitable name or reference) in Avatar's **605** surrounding, and resulting Collection of Object Representations **525** are illustrated.

As shown for example in FIG. **5**A, Object **615***a* exists in Avatar's **605** surrounding. Object **615***a* may be recognized as a person. Object **615***a* may be located at a distance of 13 m from Avatar **605**. Object **615***a* may be located at a bearing/angle of 78° from Avatar's **605** centerline. Object **615***a* may be identified as Agent Smith. Furthermore, Object **615***b* exists in Avatar's **605** surrounding. Object **615***b* may be recognized as a rock. Object **615***b* may be located at a distance of 10 m from Avatar **605**. Object **615***b* may be located at a bearing/angle of 211° from Avatar's **605** centerline. Furthermore, Object **615***c* exists in Avatar's **605** surrounding. Object **615***c* may be recognized as a robot. Object **615***c* may be located at a distance of 8 m from Avatar **605**. Object **615***c* may be located at a bearing/angle of 332° from Avatar's **605** centerline. Any Objects **615** instead of or in addition to Object **615***a*, Object **615***b*, and Object **615***c* may exist in Avatar's **605** surrounding, one or more of which can be obtained, learned, and/or used. In some designs, some Objects **615** can be omitted. Which Objects **615** or types of Objects **615** are obtained, learned, and/or used can be defined by a user, by ACAAO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. In further designs, a 3D Application Program **18** may include elevated Objects **615** such as flying objects (i.e. flying animals, aircraft, etc.), objects on hills or mountains, objects on buildings, and/or others in which case altitudinal information related to distance and bearing/angle of Objects **615** relative to Avatar **605** can be obtained, learned, and/or used. Any unit of distance and/or bearing/angle can be utilized instead of or in addition to meters and/or angular degrees.

As shown for example in FIG. **5**B, Object Processing Unit **140** may create or generate Collection of Object Representations **525** including Object Representation **625***a* representing Object **615***a*, Object Representation **625***b* representing Object **615***b*, Object Representation **625***c* representing Object **615***c*, etc. For instance, Object Representation **625***a* may include Object Property **630***aa* "Person" in Category **635***aa* "Type", Object Property **630***ab* "Agent Smith" in Category **635***ab* "Identity", Object Property **630***ac* "13 m" in Category **635***ac* "Distance", Object Property **630***ad* "78°" in Category **635***ad* "Bearing", etc. Also, Object Representation **625***b* may include Object Property **630***ba* "Rock" in Category **635***ba* "Type", Object Property **630***bb* "10 m" in Category **635***bb* "Distance", Object Property **630***bc* "211°" in Category **635***bc* "Bearing", etc. Also, Object Representation **625***c* may include Object Property **630***ca* "Robot" in Category **635***ca* "Type", Object Property **630***cb* "8 m" in Category **635***cb* "Distance", Object Property **630***cc* "332°" in Category **635***cc* "Bearing", etc. Any number of Object Representations **625**, and/or other elements or information can be included in Collection of Object Representations **525**. Any number of Object Properties **630** (also referred to simply as object properties or other suitable name or reference), and/or other elements or information can be included in an Object Representation **625**. In some aspects, a reference to Collection of Object Representations **525** comprises a reference to a collection of Object Properties **630** and/or other elements or information related to one or more Objects **615**. Other additional Object Representations **625**, Object Properties **630**, elements, and/or information can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate embodiments of Collection of Object Representations **525**.

Referring now to ACAAO Unit **100**, ACAAO Unit **100** comprises any hardware, programs, or a combination thereof. ACAAO Unit **100** comprises the functionality for learning the operation of Avatar **605** in circumstances including objects with various properties. ACAAO Unit **100** comprises the functionality for structuring and/or storing this knowledge in a knowledgebase (i.e. neural network, graph, sequences, other repository, etc.). ACAAO Unit **100** comprises the functionality for enabling autonomous operation of Avatar **605** in circumstances including objects with various properties. In some embodiments of Application Programs **18** that do not comprise Avatar **605** or rely on Avatar **605** for their operation, ACAAO Unit **100** comprises the functionality for learning the operation of Application Program **18** in circumstances including objects with various properties similar to the learning functionalities described with respect to Avatar **605**. Also, in such embodiments, ACAAO Unit **100** comprises the functionality for enabling autonomous operation of Application Program **18** in circumstances including objects with various properties similar to the autonomous operation functionalities described with respect to Avatar **605**. ACAAO Unit **100** comprises the functionality for interfacing with or attaching to Avatar **605**, Application Program **18**, Processor **11**, and/or other processing element. ACAAO Unit **100** comprises the functionality for obtaining instruction sets, data, and/or other information used, implemented, and/or executed by Avatar **605**, Application Program **18**, Processor **11**, and/or other processing element. ACAAO Unit **100** comprises the functionality for modifying instruction sets, data, and/or other information used, implemented, and/or executed by Avatar **605**, Application Program **18**, Processor **11**, and/or other processing element. ACAAO Unit **100** comprises learning, anticipating, decision making, automation, and/or other functionalities disclosed herein. Statistical, artificial intelligence, machine learning, and/or other models or techniques are utilized to implement the disclosed devices, systems, and methods. In some designs, ACAAO Unit **100** and/or elements thereof may be or include a hardware element embedded or built into Processor **11**, and/or other processing element. In other designs, ACAAO Unit **100** and/or elements thereof may be or include a hardware element coupled to or interfaced with Avatar **605**, Application Program **18**, Processor **11**, and/or other processing element. In other designs, ACAAO Unit **100** and/or elements thereof may be or include a program operating on Processor **11**, and/or other processing element. In further designs, ACAAO Unit **100** and/or elements thereof may be or include a program coupled to or interfaced with Avatar **605**, Application Program **18**, Processor **11**, and/or other processing element. In further designs, ACAAO Unit **100** and/or elements thereof may be or include a program embedded or built into Avatar **605**, Application Program **18**, and/or other processing element. ACAAO Unit **100** can be provided in a combination of the aforementioned or other suitable configurations in alternate designs.

When ACAAO Unit **100** functionalities are applied to Avatar **605**, Application Program **18**, Processor **11**, and/or other processing element, Avatar **605**, Application Program **18**, Processor **11**, and/or other processing element may become autonomous. ACAAO Unit **100** may take control from, share control with, and/or release control to Avatar **605**, Application Program **18**, Processor **11**, and/or other processing element to implement autonomous operation of Avatar **605**, Application Program **18**, Processor **11**, and/or

other processing element. ACAAO Unit **100** may take control from, share control with, and/or release control to Avatar **605**, Application Program **18**, Processor **11**, and/or other processing element automatically or after prompting User **50** to allow it. In some aspects, Avatar **605**, Application Program **18**, Processor **11**, and/or other processing element may include or be provided with anticipatory (also referred to as alternate or other suitable name or reference) instructions or instruction sets that User **50** did not issue or cause to be executed. Such anticipatory instructions or instruction sets include instruction sets that User **50** may want or is likely to issue or cause to be executed. Anticipatory instructions or instruction sets can be generated by ACAAO Unit **100** or elements thereof based on circumstances including objects with various properties. As such, Avatar **605**, Application Program **18**, Processor **11**, and/or other processing element may include or be provided with some or all original instructions or instruction sets and/or any anticipatory instructions or instruction sets generated by ACAAO Unit **100**. Therefore, autonomous operating of Avatar **605**, Application Program **18**, Processor **11**, and/or other processing element may include executing some or all original instructions or instruction sets and/or any anticipatory instructions or instruction sets generated by ACAAO Unit **100**.

In one example, ACAAO Unit **100** can overwrite or rewrite the original instructions or instruction sets with ACAAO Unit **100**-generated instructions or instruction sets. In another example, ACAAO Unit **100** can insert or embed ACAAO Unit **100**-generated instructions or instruction sets among the original instructions or instruction sets. In a further example, ACAAO Unit **100** can branch, redirect, or jump to ACAAO Unit **100**-generated instructions or instruction sets from the original instructions or instruction sets.

In some embodiments, autonomous Avatar **605** operating comprises determining, by ACAAO Unit **100**, a next instruction or instruction set to be executed based on Avatar's **605** circumstances including objects with various properties prior to the user issuing or causing to be executed the next instruction or instruction set. In other embodiments, autonomous Avatar **605** operating comprises determining, by ACAAO Unit **100**, a next instruction or instruction set to be executed based on Avatar's **605** circumstances including objects with various properties prior to the system receiving the next instruction or instruction set.

In some embodiments, autonomous Avatar **605** operating includes a partially or fully autonomous operating. In an example involving partially autonomous Avatar **605** operating, a user confirms ACAAO Unit **100**-generated instructions or instruction sets prior to their execution. In an example involving fully autonomous application operating, ACAAO Unit **100**-generated instructions or instruction sets are executed without user or other system confirmation (i.e. automatically, etc.).

In some embodiments, a combination of ACAAO Unit **100** and other systems and/or techniques can be utilized to implement Avatar's **605** operation. In one example, ACAAO Unit **100** may be a primary or preferred system for implementing Avatar's **605** operation. While operating autonomously under the control of ACAAO Unit **100**, Avatar **605** may encounter a circumstance including objects with various properties that has not been encountered or learned before. In such situations, User **50** and/or non-ACAAO system may take control of Avatar's **605** operation. ACAAO Unit **100** may take control again when Avatar **605** encounters a previously learned circumstance including objects with various properties. Naturally, ACAAO Unit **100** can learn Avatar's **605** operation in circumstances while User **50** and/or non-ACAAO system is in control of Avatar **605**, thereby reducing or eliminating the need for future involvement of User **50** and/or non-ACAAO system. In another example, User **50** and/or non-ACAAO system may be a primary or preferred system for implementing Avatar's **605** operation. While operating under the control of User **50** and/or non-ACAAO system, User **50** and/or non-ACAAO system may release control to ACAAO Unit **100** for any reason (i.e. User **50** gets tired or distracted, non-ACAAO system gets stuck or cannot make a decision, etc.), at which point Avatar **605** can be controlled by ACAAO Unit **100**. In some designs, ACAAO Unit **100** may take control in certain special circumstances including objects with various properties where ACAAO Unit **100** may offer superior performance even though User **50** and/or non-ACAAO system may generally be preferred. Once Avatar **605** leaves such special circumstances, ACAAO Unit **100** may release control to User **50** and/or non-ACAAO system. In general, ACAAO Unit **100** can take control from, share control with, or release control to User **50**, non-ACAAO system, and/or other system or process at any time, in any circumstances, and remain in control for any period of time as needed.

In some embodiments, ACAAO Unit **100** may control one or more elements of Avatar **605** while User **50** and/or non-ACAAO system may control other one or more elements of Avatar **605**. For example, ACAAO Unit **100** may control Avatar's **605** movement, while User **50** and/or non-ACAAO system may control Avatar's **605** aiming and shooting. Any other combination of controlling various elements or functions of Avatar **605** by ACAAO Unit **100**, User **50**, and/or non-ACAAO system can be implemented.

In some embodiments, ACAAO Unit **100** enables learning of a particular User's **50** knowledge, methodology, or style of operating Avatar **605**. In some aspects, learning of a particular User's **50** knowledge, methodology, or style of operating Avatar **605** includes learning the User's **50** directing or operating Avatar **605** in circumstances including objects with various properties. In one example, one User **50** may shoot an opponent while another User **50** may strike the opponent with a sword in a computer game. In another example, one User **50** may jump over an obstacle while another User **50** may move around the obstacle in a virtual world application. In a further example, one User **50** may drive fast while another User **50** may drive cautiously in a racing game, and so on. The knowledge of User's **50** methodology or style of operating Avatar **605** can be used to enable personalized autonomous operation of Avatar **605** specific to a particular User **50**. Therefore, ACAAO-enabled Avatar **605** may exemplify User's **50** knowledge, methodology, or style of operating Avatar **605** as learned from User **50**. In some designs, this functionality enables one or more ACAAO-enabled Avatars **605** to be utilized in Application Program **18** (i.e. computer game, virtual world, etc.) to assist User **50** in defeating an opponent or achieving another goal. For example, User **50** can utilize a team of ACAAO-enabled Avatars **605** each of which may exemplify User's **50** knowledge, methodology, or style of operating Avatar **605**. In other designs, ACAAO Unit **100** enables a professional or other experienced Application Program **18** operator (i.e. game player, etc.) to record his/her knowledge, methodology, or style of operating Avatar **605** into Knowledgebase **530** (i.e. Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.) and/or other repository. User **50** can then sell or make available his/her knowledge, methodology, or style of operating Avatar **605** to other users who may want to implement User's **50** knowledge, methodology, or style of operating

Avatar **605**. Knowledgebase **530** and/or other repository comprising User's **50** knowledge, methodology, or style of operating Avatar **605** can be available to other users via a storage medium, via a network, or via other means. In some implementations, User's **50** knowledge, methodology, or style of operating Avatar **605** can be applied to or implemented on any Object **615** of Application Program **18** as applicable, thereby enabling any Object **615** to exemplify User's **50** knowledge, methodology, or style of operating as learned from User **50**. For example, a computer game developer may associate Knowledgebase **530** comprising User's **50** knowledge, methodology, or style of operating Avatar **605** with an Object **615** (i.e. tank, robot, aircraft, etc.), thereby enabling the Object **615** to operate based on the knowledge in the Knowledgebase **530**.

Referring now to Acquisition Interface **120**, Acquisition Interface **120** comprises the functionality for obtaining and/or receiving instruction sets, data, and/or other information. Acquisition Interface **120** comprises the functionality for obtaining and/or receiving instruction sets, data, and/or other information related to the operation of Avatar **605**, Application Program **18**, Processor **11**, and/or other processing element. Acquisition Interface **120** comprises the functionality for obtaining and/or receiving instruction sets, data, and/or other information from Avatar **605**, Application Program **18**, Processor **11**, and/or other processing element. Acquisition Interface **120** comprises the functionality for obtaining and/or receiving instruction sets, data, and/or other information at runtime. In one example, Acquisition Interface **120** can obtain Instruction Sets **526** used or executed in operating Avatar **605** operation, and transmit the Instruction Sets **526** to Artificial Intelligence Unit **110** for learning Avatar's **605** operation in circumstances including objects with various properties. In another example, in Application Programs **18** that do not comprise Avatar **605** or do not rely on Avatar **605** for their operation, Acquisition Interface **120** can obtain Instruction Sets **526** used or executed in operating Application Program **18**, and transmit the Instruction Sets **526** to Artificial Intelligence Unit **110** for learning Application Program's **18** operation in circumstances including objects with various properties. Acquisition Interface **120** also comprises the functionality for tracing or profiling of Avatar **605**, Application Program **18**, Processor **11**, and/or other processing element. Tracing or profiling may include adding trace code or instrumentation to Avatar **605** (i.e. Avatar's **605** object code, etc.) or Application Program **18**, and/or outputting trace information (i.e. instruction sets, data, and/or other information, etc.) to a receiving target. Acquisition Interface **120** further comprises the functionality for attaching to or interfacing with Avatar **605**, Application Program **18**, Processor **11**, and/or other processing element. In some aspects, Acquisition Interface **120** can access and/or read runtime engine/environment, virtual machine, operating system, compiler, interpreter, translator, execution stack, file, object, data structure, and/or other computing system elements. In other aspects, Acquisition Interface **120** can access and/or read memory, storage, and/or other repository. In further aspects, Acquisition Interface **120** can access and/or read Processor **11** registers and/or other Processor **11** elements. In further aspects, Acquisition Interface **120** can access and/or read inputs and/or outputs of Avatar **605**, Application Program **18**, Processor **11**, and/or other processing element. In further aspects, Acquisition Interface **120** can access and/or read functions, methods, procedures, routines, subroutines, and/or other elements of Avatar **605** and/or Application Program **18**. In further aspects, Acquisition Interface **120** can access and/or read source code, bytecode, compiled, interpreted code, translated code, machine code, and/or other code. In further aspects, Acquisition Interface **120** can access and/or read values, variables, parameters, and/or other data or information. Acquisition Interface **120** also comprises the functionality for transmitting the obtained instruction sets, data, and/or other information to Artificial Intelligence Unit **110** and/or other elements. As such, Acquisition Interface **120** provides input into Artificial Intelligence Unit **110** for knowledge structuring, anticipating, decision making, and/or other functionalities later in the process. Acquisition Interface **120** also comprises other disclosed functionalities.

Acquisition Interface **120** can employ various techniques for obtaining instruction sets, data, and/or other information. In one example, Acquisition Interface **120** can attach to and/or obtain Avatar's **605**, Application Program's **18**, Processor's **11**, and/or other processing element's instruction sets, data, and/or other information through tracing or profiling techniques. Tracing or profiling may be used for outputting Avatar's **605**, Application Program's **18**, Processor's **11**, and/or other processing element's instruction sets, data, and/or other information at runtime. For instance, tracing or profiling may include adding trace code (i.e. instrumentation, etc.) to Avatar **605** (i.e. Avatar's **605** object code, etc.) or Application Program **18**, and/or outputting trace information to a specific target. The outputted trace information (i.e. instruction sets, data, and/or other information, etc.) can then be provided to or recorded into a file, data structure, repository, application, and/or other system or target that may receive such trace information. As such, Acquisition Interface **120** can utilize tracing or profiling to obtain instruction sets, data, and/or other information and provide them as input into Artificial Intelligence Unit **110**. In some aspects, instrumentation can be performed in source code, bytecode, compiled code, interpreted code, translated code, machine code, and/or other code. In other aspects, instrumentation can be performed in various elements of a computing system such as memory, virtual machine, runtime engine/environment, operating system, compiler, interpreter, translator, processor registers, and/or other elements. In yet other aspects, instrumentation can be performed in various abstraction layers of a computing system such as in software layer (i.e. Application Program **18**, etc.), in virtual machine (if VM is used), in operating system, in Processor **11**, and/or in other layers or areas that may exist in a particular computing system implementation. In yet other aspects, instrumentation can be performed at various times in Avatar's **605** or Application Program's **18** execution such as source code write time, compile time, interpretation time, translation time, linking time, loading time, runtime, and/or others. In yet other aspects, instrumentation can be performed at various granularities or code segments such as some or all lines of code, some or all statements, some or all instructions or instruction sets, some or all basic blocks, some or all functions/routines/subroutines, and/or some or all other code segments.

In some embodiments, Avatar **605** (i.e. Avatar's **605** object code, etc.) or Application Program **18** can be automatically instrumented. For example, Acquisition Interface **120** can access Avatar's **605** or Application Program's **18** source code, bytecode, or machine code and select instrumentation points of interest. Selecting instrumentation points may include finding locations in the source code, bytecode, or machine code corresponding to function calls, function entries, function exits, object creations, object destructions, event handler calls, new lines (i.e. to instrument all lines of code, etc.), thread creations, throws, and/or

other points of interest. Instrumentation code can then be inserted at the instrumentation points of interest to output Avatar's **605** or Application Program's **18** instruction sets, data, and/or other information. In response to executing instrumentation code, Avatar's **605** or Application Program's **18** instruction sets, data, and/or other information may be received by Acquisition Interface **120**. In some aspects, Avatar's **605** or Application Program's **18** source code, bytecode, or machine code can be dynamically instrumented. For example, instrumentation code can be dynamically inserted into Avatar **605** (i.e. Avatar's **605** object code, etc.) or Application Program **18** at runtime.

In other embodiments, Avatar **605** or Application Program **18** can be manually instrumented. In one example, a programmer can instrument a function call by placing an instrumenting function such as traceAvatar( ), traceApplication( ), etc. immediately after the function call as in the following example.

```
Avatar.moveForward(12);
traceAvatar('Avatar.moveForward(12);');
```

In another example, an instrumenting function can be placed immediately before the function call, or at the beginning, end, or anywhere within the function itself. A programmer may instrument all function calls or only function calls of interest. In a further example, a programmer can instrument all lines of code or only code lines of interest. In a further example, a programmer can instrument other elements utilized or implemented within Avatar **605** or Application Program **18** such as objects and/or any of their functions, data structures and/or any of their functions, event handlers and/or any of their functions, threads and/or any of their functions, and/or other elements or functions. Similar instrumentation as in the preceding examples can be performed automatically or dynamically. In some designs where manual code instrumentation is utilized, Acquisition Interface **120** can optionally be omitted and Avatar's **605** or Application Program's **18** instruction sets, data, and/or other information may be transmitted directly to Artificial Intelligence Unit **110**.

In some embodiments, ACAAO Unit **100** can be selective in learning instruction sets, data, and/or other information to those implemented, utilized, or related to an object, data structure, repository, thread, function, and/or other element of Application Program **18**. In some aspects, Acquisition Interface **120** can obtain Application Program's **18** instruction sets, data, and/or other information implemented, utilized, or related to a certain object in an object oriented Application Program **18**. For example, Acquisition Interface **120** can obtain Application Program's **18** instruction sets, data, and/or other information implemented, utilized, or related to Avatar **605**.

In some embodiments, various computing systems and/or platforms may provide native tools for obtaining instruction sets, data, and/or other information. Also, independent vendors may provide portable tools with similar functionalities that can be utilized across different computing systems and/or platforms. These native and portable tools may provide a wide range of functionalities such as instrumentation, tracing or profiling, logging application or system messages, outputting custom text messages, outputting objects or data structures, outputting functions/routines/subroutines or their invocations, outputting variable or parameter values, outputting thread or process behaviors, outputting call or other stacks, outputting processor registers, providing runtime memory access, providing inputs and/or outputs, performing live application monitoring, and/or other capabilities. One of ordinary skill in art will understand that, while all possible variations of the techniques to obtain instruction sets, data, and/or other information are too voluminous to describe, these techniques are within the scope of this disclosure.

In one example, obtaining Avatar's **605** and/or Application Program's **18** instruction sets, data, and/or other information can be implemented through the .NET platform's native tools for application tracing or profiling such as System.Diagnostics.Trace, System.Diagnostics.Debug, or System.Diagnostics.TraceSource classes for tracing execution flow, and/or System.Diagnostics.Process, System.Diagnostics.EventLog, or System.Diagnostics.PerformanceCounter classes for profiling code, accessing local and remote processes, starting and stopping system processes, and interacting with Windows event logs, etc. For instance, a set of trace switches can be created that output an application's information. The switches can be configured using the .config file. For a web application, this may typically be web.config file associated with the project. In a Windows application, this file may typically be named applicationName.exe.config. Trace code can be added to application code automatically or manually as previously described. Appropriate listener can be created where the trace output is received. Trace code can output trace messages to a specific target such as a file, a log, a database, an object, a data structure, and/or other repository or system. Acquisition Interface **120** or Artificial Intelligence Unit **110** can then read or obtain the trace information from these targets. In some aspects, trace code can output trace messages directly to Acquisition Interface **120**. In other aspects, trace code can output trace messages directly to Artificial Intelligence Unit **110**. In the case of outputting trace messages to Acquisition Interface **120** or directly to Artificial Intelligence Unit **110**, custom listeners can be built to accommodate these specific targets. Other platforms, tools, and/or techniques can provide equivalent or similar functionalities as the above described ones.

In another example, obtaining Avatar's **605** and/or Application Program's **18** instruction sets, data, and/or other information can be implemented through the .NET platform's Profiling API that can be used to create a custom profiler application for tracing, monitoring, interfacing with, and/or managing a profiled application. The Profiling API provides an interface that includes methods to notify the profiler of events in the profiled application. The Profiling API may also provide an interface to enable the profiler to call back into the profiled application to obtain information about the state of the profiled application. The Profiling API may further provide call stack profiling functionalities. Call stack (also referred to as execution stack, control stack, runtime stack, machine stack, the stack, etc.) includes a data structure that can store information about active subroutines of an application. The Profiling API may provide a stack snapshot method, which enables a trace of the stack at a particular point in time. The Profiling API may also provide a shadow stack method, which tracks the call stack at every instant. A shadow stack can obtain function arguments, return values, and information about generic instantiations. A function such as Function Enter can be utilized to notify the profiler that control is being passed to a function and can provide information about the stack frame and function arguments. A function such as FunctionLeave can be utilized to notify the profiler that a function is about to return to the caller and can provide information about the stack frame and function return value. An alternative to call stack profiling includes call stack sampling in which the profiler can periodically examine the stack. In some aspects, the Profil-

ing API enables the profiler to change the in-memory code stream for a routine before it is just-in-time (JIT) compiled where the profiler can dynamically add instrumentation code to all or particular routines of interest. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining Avatar's **605** and/or Application Program's **18** instruction sets, data, and/or other information can be implemented through Java platform's APIs for application tracing or profiling such as Java Virtual Machine Profiling Interface (JVMPI), Java Virtual Machine Tool Interface (JVMTI), and/or other APIs or tools. These APIs can be used for instrumentation of an application, for notification of Java Virtual Machine (VM) events, and/or other functionalities. One of the tracing or profiling techniques that can be utilized includes bytecode instrumentation. The profiler can insert bytecodes into all or some of the classes. In application execution profiling, for example, these bytecodes may include methodEntry and methodExit calls. In memory profiling, for example, the bytecodes may be inserted after each new or after each constructor. In some aspects, insertion of instrumentation bytecode can be performed either by a post-compiler or a custom class loader. An alternative to bytecode instrumentation includes monitoring events generated by the JVMPI or JVMTI interfaces. Both APIs can generate events for method entry/exit, object allocation, and/or other events. In some aspects, JVMTI can be utilized for dynamic bytecode instrumentation where insertion of instrumentation bytecodes is performed at runtime. The profiler may insert the necessary instrumentation when a selected class is invoked in an application. This can be accomplished using the JVMTI's redefineClasses method, for example. This approach also enables changing of the level of profiling as the application is running. If needed, these changes can be made adaptively without restarting the application. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining Avatar's **605** and/or Application Program's **18** instruction sets, data, and/or other information can be implemented through JVMTI's programming interface that enables creation of software agents that can monitor and control a Java application. An agent may use the functionality of the interface to register for notification of events as they occur in the application, and to query and control the application. A JVMTI agent may use JVMTI functions to extract information from a Java application. A JVMTI agent can be utilized to obtain an application's runtime information such as method calls, memory allocation, CPU utilization, and/or other information. JVMTI may include functions to obtain information about variables, fields, methods, classes, and/or other information. JVMTI may also provide notification for numerous events such as method entry and exit, exception, field access and modification, thread start and end, and/or other events. Examples of JVMTI built-in methods include GetMethodName to obtain the name of an invoked method, GetThreadInfo to obtain information for a specific thread, GetClassSignature to obtain information about the class of an object, GetStackTrace to obtain information about the stack including information about stack frames, and/or other methods. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining Avatar's **605** and/or Application Program's **18** instruction sets, data, and/or other information can be implemented through java.lang.Runtime class that provides an interface for application tracing or

profiling. Examples of methods provided in java.lang.Runtime that can be used to obtain an application's instruction sets, data, and/or other information include tracemethodcalls, traceinstructions, and/or other methods. These methods prompt the Java Virtual Machine to output trace information for a method or instruction in the virtual machine as it is executed. The destination of trace output may be system dependent and include a file, a listener, and/or other destinations where Acquisition Interface **120**, Artificial Intelligence Unit **110**, and/or other disclosed elements can access needed information. In addition to tracing or profiling tools native to their respective computing systems and/or platforms, many independent tools exist that provide tracing or profiling functionalities on more than one computing system and/or platform. Examples of these tools include Pin, DynamoRIO, KernInst, DynInst, Kprobes, OpenPAT, DTrace, SystemTap, and/or others. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining Avatar's **605** and/or Application Program's **18** instruction sets, data, and/or other information can be implemented through logging tools of the platform and/or operating system on which an application runs. Some logging tools may include nearly full feature sets of the tracing or profiling tools previously described. In one example, Visual Basic enables logging of runtime messages through its Microsoft.VisualBasic.Logging namespace that provides a log listener where the log listener may direct logging output to a file and/or other target. In another example, Java enables logging through its java.util.logging class. In some aspects, obtaining an application's instruction sets, data, and/or other information can be implemented through logging capabilities of the operating system on which an application runs. For example, Windows NT features centralized log service that applications and operating-system components can utilize to report their events including any messages. Windows NT provides functionalities for system, application, security, and/or other logging. An application log may include events logged by applications. Windows NT, for example, may include support for defining an event source (i.e. application that created the event, etc.). Windows Vista, for example, supports a structured XML log-format and designated log types to allow applications to more precisely log events and to help interpret the events. Examples of different types of event logs include administrative, operational, analytic, debug, and/or other log types including any of their subcategories. Examples of event attributes that can be utilized include eventID, level, task, opcode, keywords, and/or other event attributes. Windows wevtutil tool enables access to events, their structures, registered event publishers, and/or their configuration even before the events are fired. Wevtutil supports capabilities such as retrieval of the names of all logs on a computing device; retrieval of configuration information for a specific log; retrieval of event publishers on a computing device; reading events from an event log, from a log file, or using a structured query; exporting events from an event log, from a log file, or using a structured query to a specific target; and/or other capabilities. Operating system logs can be utilized solely if they contain sufficient information on an application's instruction sets, data, and/or other information. Alternatively, operating system logs can be utilized in combination with another source of information (i.e. trace information, call stack, processor registers, memory, etc.) to reconstruct the application's instruction sets, data, and/or other information needed for Artificial Intelligence Unit **110** and/or other elements. In addition to

logging capabilities native to their respective platforms and/or operating systems, many independent tools exist that provide logging on different platforms and/or operating systems. Examples of these tools include Log4j, Logback, SmartInspect, NLog, log4net, Microsoft Enterprise Library, ObjectGuy Framework, and/or others. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining Avatar's **605** and/or Application Program's **18** instruction sets, data, and/or other information can be implemented through tracing or profiling the operating system on which an application runs. As in tracing or profiling an application, one of the techniques that can be utilized includes adding instrumentation code to the operating system's source code. Such instrumentation code can be added to the operating system's source code before kernel compilation or recompilation, for instance. This type of instrumentation may involve defining or finding locations in the operating system's source code where instrumentation code may be inserted. Kernel instrumentation can also be performed without the need for kernel recompilation or rebooting. In some aspects, instrumentation code can be added at locations of interest through binary rewriting of compiled kernel code. In other aspects, kernel instrumentation can be performed dynamically where instrumentation code is added and/or removed where needed at runtime. For instance, dynamic instrumentation may overwrite kernel code with a branch instruction that redirects execution to instrumentation code or instrumentation routine. In yet other aspects, kernel instrumentation can be performed using just-in-time (JIT) dynamic instrumentation where execution may be redirected to a copy of kernel's code segment that includes instrumentation code. This type of instrumentation may include a JIT compiler and creation of a copy of the original code segment having instrumentation code or calls to instrumentation routines embedded into the original code segment. Instrumentation of the operating system may enable total system visibility including visibility into an application's behavior by enabling generation of low level trace information. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining Avatar's **605** and/or Application Program's **18** instruction sets, data, and/or other information can be implemented through tracing or profiling the processor on which an application runs. For example, some Intel processors provide Intel Processor Trace (i.e. Intel PT, etc.), a low-level tracing feature that enables recording executed instruction sets, and/or other data or information of one or more applications. Intel PT is facilitated by the Processor Trace Decoder Library along with its related tools. Intel PT is a low-overhead execution tracing feature that records information about application execution on each hardware thread using dedicated hardware facilities. The recorded execution/trace information is collected in data packets that can be buffered internally before being sent to a memory subsystem or another system or element (i.e. Acquisition Interface **120**, Artificial Intelligence Unit **110**, etc.). Intel PT also enables navigating the recorded execution/trace information via reverse stepping commands. Intel PT can be included in an operating system's core files and provided as a feature of the operating system. Intel PT can trace globally some or all applications running on an operating system. Acquisition Interface **120** or Artificial Intelligence Unit **110** can read or obtain the recorded execution/trace information from Intel PT. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining Avatar's **605** and/or Application Program's **18** instruction sets, data, and/or other information can be implemented through branch tracing or profiling. Branch tracing may include an abbreviated instruction trace in which only the successful branch instruction sets are traced or recorded. Branch tracing can be implemented through utilizing dedicated processor commands, for example. Executed branches may be saved into special branch trace store area of memory. With the availability and reference to a compiler listing of the application together with branch trace information, a full path of executed instruction sets can be reconstructed. The full path can also be reconstructed with a memory dump (containing the program storage) and branch trace information. In some aspects, branch tracing can be utilized for pre-learning or automated learning of an application's instruction sets, data, and/or other information where a number of application simulations (i.e. simulations of likely/common operations, etc.) are performed. As such, the application's operation can be learned automatically saving the time that would be needed to learn the application's operation directed by a user. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining Avatar's **605** and/or Application Program's **18** instruction sets, data, and/or other information can be implemented through assembly language. Assembly language is a low-level programming language for a computer or other programmable device in which there is a strong correlation between the language and the architecture's machine instruction sets. Syntax, addressing modes, operands, and/or other elements of an assembly language instruction set may translate directly into numeric (i.e. binary, etc.) representations of that particular instruction set. Because of this direct relationship with the architecture's machine instruction sets, assembly language can be a powerful tool for tracing or profiling an application's execution in processor registers, memory, and/or other computing system components. For example, using assembly language, memory locations of a loaded application can be accessed, instrumented, and/or otherwise manipulated. In some aspects, assembly language can be used to rewrite or overwrite original in-memory instruction sets of an application with instrumentation instruction sets. In other aspects, assembly language can be used to redirect application's execution to instrumentation routine/subroutine or other code segment elsewhere in memory by inserting a jump into the application's in-memory code, by redirecting program counter, or by other techniques. Some operating systems may implement protection from changes to applications loaded into memory. Operating system, processor, or other low level commands such as Linux mprotect command or similar commands in other operating systems may be used to unprotect the protected locations in memory before the change. In yet other aspects, assembly language can be used to obtain instruction sets, data, and/or other information through accessing and/or reading instruction register, program counter, other processor registers, memory locations, and/or other components of a computing system. In yet other aspects, high-level programming languages may call or execute an external assembly language program to facilitate obtaining instruction sets, data, and/or other information as previously described. In yet other aspects, relatively low-level programming languages such as C may allow embedding assembly language directly in their source code such as,

for example, using asm keyword of C. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, it may be sufficient to obtain user or other inputs, variables, parameters, and/or other data in some procedural, simple object oriented, or other applications. In one instance, a simple procedural application executes a sequence of instruction sets until the end of the program. During its execution, the application may receive user or other input, store the input in a variable, and perform calculations using the variable to reach a result. The value of the variable can be obtained or traced. In another instance, a more complex procedural application comprises one or more functions/routines/subroutines each of which may include a sequence of instruction sets. The application may execute a main sequence of instruction sets with a branch to a function/routine/subroutine. During its execution, the application may receive user or other input, store the input in a variable, and pass the variable as a parameter to the function/routine/subroutine. The function/routine/subroutine may perform calculations using the parameter and return a value that the rest of the application can use to reach a result. The value of the variable or parameter passed to the function/routine/subroutine, and/or return value can be obtained or traced. Values of user or other inputs, variables, parameters, and/or other items of interest can be obtained through previously described tracing, instrumentation, and/or other techniques. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

Referring to FIG. **6**, in yet another example, obtaining Avatar's **605** and/or Application Program's **18** instruction sets, data, and/or other information may be implemented through tracing, profiling, or sampling of instruction sets or data in processor registers, memory, or other computing system components where instruction sets, data, and/or other information may be stored or utilized. For example, Instruction Register **212** may be part of Processor **11** and it may store the instruction set currently being executed or decoded. In some processors, Program Counter **211** (also referred to as instruction pointer, instruction address register, instruction counter, or part of instruction sequencer, etc.) may be incremented after fetching an instruction set, and it may hold or point to the memory address of the next instruction set to be executed. In a processor where the incrementation precedes the fetch, Program Counter **211** may point to the current instruction set being executed. In the instruction cycle, an instruction set may be loaded into Instruction Register **212** after Processor **11** fetches it from location in Memory **12** pointed to by Program Counter **211**. Instruction Register **212** may hold the instruction set while it is decoded by Instruction Decoder **213**, prepared, and executed. In some aspects, data (i.e. operands, etc.) needed for instruction set execution may be loaded from Memory **12** into a register within Register Array **214**. In other aspects, the data may be loaded directly into Arithmetic Logic Unit **215**. For instance, as instruction sets pass through Instruction Register **212** during application execution, they may be transmitted to Acquisition Interface **120** as shown. Examples of the steps in execution of a machine instruction set may include decoding the opcode (i.e. portion of a machine instruction set that may specify the operation to be performed), determining where the operands may be located (depending on architecture, operands may be in registers, the stack, memory, I/O ports, etc.), retrieving the operands, allocating processor resources to execute the instruction set (needed in some types of processors), performing the operation indicated by the instruction set, saving the results of execution, and/or other execution steps. Examples of the types of machine instruction sets that can be utilized include arithmetic, data handling, logical, program control, as well as special and/or other instruction set types. In addition to the ones described or shown, examples of other computing system or processor components that can be used during an instruction cycle include memory address register (MAR) that may hold the address of a memory block to be read from or written to; memory data register (MDR) that may hold data fetched from memory or data waiting to be stored in memory; data registers that may hold numeric values, characters, small bit arrays, or other data; address registers that may hold addresses used by instruction sets that indirectly access memory; general purpose registers (GPRs) that may store both data and addresses; conditional registers that may hold truth values often used to determine whether some instruction set should or should not be executed; floating point registers (FPRs) that may store floating point numbers; constant registers that may hold read-only values such as zero, one, or pi; special purpose registers (SPRs) such as status register, program counter, or stack pointer that may hold information on program state; machine-specific registers that may store data and settings related to a particular processor; Register Array **214** that may include an array of any number of processor registers; Arithmetic Logic Unit **215** that may perform arithmetic and logic operations; control unit that may direct processor's operation; and/or other circuits or components. Tracing, profiling, or sampling of processor registers, memory, or other computing system components can be implemented in a program, combination of hardware and program, or purely hardware system. Dedicated hardware may be built to perform tracing, profiling, or sampling of processor registers or any computing system components with marginal or no impact to computing overhead.

One of ordinary skill in art will recognize that FIG. **6** depicts one of many implementations of processor or computing system components, and that various additional components can be included, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate implementations. Processor or computing system components may be arranged or connected differently in alternate implementations. Processor or computing system components may also be connected with external elements using various connections. For instance, the connection between Instruction Register **212** and Acquisition Interface **120** may include any number or types of connections such as, for example, a dedicated connection for each bit of Instruction Register **212** (i.e. 32 connections for a 32 bit Instruction Register **212**, etc.). Any of the described or other connections or interfaces may be implemented among any processor or computing system components and Acquisition Interface **120** or other elements.

Other additional techniques or elements can be utilized as needed for obtaining instruction sets, data, and/or other information, or some of the disclosed techniques or elements can be excluded, or a combination thereof can be utilized in alternate embodiments. As an avatar (i.e. Avatar **605**, etc.) may be part of an application (i.e. Application Program **18**, etc.), it should be noted that obtaining an avatar's instruction sets, data, and/or other information may include same or similar techniques as the aforementioned obtaining an application's instruction sets, data, and/or other information, and vice versa.

Referring to FIGS. **7A-7E**, some embodiments of Instruction Sets **526** are illustrated. In some aspects, Instruction Set

526 includes one or more instructions or commands related to Avatar **605**. For example, Instruction Set **526** may include one or more instructions or commands for operating Avatar **605**. In other aspects, Instruction Set **526** includes one or more instructions or commands of Application Program **18**. For example, Instruction Set **526** may include one or more instructions or commands for operating Application Program **18**. In further aspects, Instruction Set **526** includes one or more inputs into and/or outputs from Avatar **605**, Application Program **18**, Processor **11**, and/or other processing element. In further aspects, Instruction Set **526** includes one or more values or states of registers and/or other components of Processor **11** and/or other processing element. In general, Instruction Set **526** may include one or more instructions, commands, keywords, symbols (i.e. parentheses, brackets, commas, semicolons, etc.), operators (i.e. =, <, >, etc.), variables, values, objects, data structures, functions (i.e. Function1( ), FIRST( ), MIN( ), SQRT( ), etc.), parameters, states, signals, inputs, outputs, characters, digits, references thereto, and/or other components. Therefore, the terms instruction set, command, instruction, signal, or other such terms may be used interchangeably herein depending on context.

In an embodiment shown in FIG. **7**A, Instruction Set **526** includes code of a high-level programming language (i.e. Java, C++, etc.) comprising the following function call construct: Function1 (Parameter1, Parameter2, Parameter3, . . .). An example of a function call applying the above construct includes the following Instruction Set **526**: moveTo(Avatar, 11, 23). The function or reference thereto "moveTo(Avatar, 11, 23)" may be an Instruction Set **526** directing Avatar **605** to move to a location with coordinates 11 and 23, for example. In another embodiment shown in FIG. **7**B, Instruction Set **526** includes structured query language (SQL). In a further embodiment shown in FIG. **7**C, Instruction Set **526** includes bytecode (i.e. Java bytecode, Python bytecode, CLR bytecode, etc.). In a further embodiment shown in FIG. **7**D, Instruction Set **526** includes assembly code. In a further embodiment shown in FIG. **7**E, Instruction Set **526** includes machine code. Instruction Set **526** may include any other language or construct in alternate embodiments.

Referring to FIGS. **8**A-**88**B, some embodiments of Extra Information **527** (also referred to as Extra Info **527**) are illustrated. In an embodiment shown in FIG. **8**A, Collection of Object Representations **525** may include or be associated with Extra Info **527**. In an embodiment shown in FIG. **8**B, Instruction Set **526** may include or be associated with Extra Info **527**.

Extra Info **527** comprises the functionality for storing any information useful in comparisons or decision making performed in autonomous Avatar **605** operation, and/or other functionalities. One or more Extra Infos **527** can be stored in, appended to, or associated with a Collection of Object Representations **525**, Instruction Set **526**, and/or other element. In some embodiments, the system can obtain Extra Info **527** at a time of creating or generating Collection of Object Representations **525**. In other embodiments, the system can obtain Extra Info **527** at a time of acquiring Instruction Set **526**. In general, Extra Info **527** can be obtained at any time. Examples of Extra Info **527** include time information, location information, computed information, visual information, acoustic information, contextual information, and/or other information. Any information can be utilized that can provide information for enhanced comparisons or decision making performed in autonomous Avatar **605** operation. Which information is utilized and/or stored in Extra Info **527** can be set by a user, by ACAAO system administrator, or automatically by the system. Extra Info **527** may include or be referred to as contextual information, and vice versa. Therefore, these terms may be used interchangeably herein depending on context.

In some aspects, time information (i.e. time stamp, etc.) can be utilized and/or stored in Extra Info **527**. Time information can be useful in comparisons or decision making performed in autonomous Avatar **605** operation related to a specific time period. For example, Extra Info **527** may include time information related to when Avatar **605** performed an operation. Time information can be obtained from the system clock, online clock, oscillator, or other time source. In other aspects, location information (i.e. coordinates, etc.) can be utilized and/or stored in Extra Info **527**. Location information can be useful in comparisons or decision making performed in autonomous Avatar **605** operation related to a specific place. For example, Extra Info **527** may include location information related to where Avatar **605** performed an operation. Location information can be obtained from Application Program's **18** engine (i.e. game engine in which the game is implemented, etc.), runtime environment, functions for providing location information on objects, and/or others as previously described. In further aspects, computed information can be utilized and/or stored in Extra Info **527**. Computed information can be useful in comparisons or decision making performed in autonomous Avatar **605** operation where information can be calculated, inferred, or estimated from other available information. ACAAO Unit **100** may include computational functionalities to create Extra Info **527** by performing calculations, inferences, or estimations using other information. In one example, Avatar's **605** direction of movement can be computed or estimated using Avatar's **605** location information. In another example, Avatar's **605** speed can be computed or estimated using Avatar's **605** location and/or time information. In a further example, speeds, directions of movement, trajectories, distances, and/or other properties of objects around Avatar **605** can similarly be computed or estimated, thereby providing geo-spatial and situational awareness and/or capabilities to Avatar **605**. ACAAO Unit **100** can utilize geometry, trigonometry, Pythagorean theorem, and/or other theorems, formulas, or disciplines in its calculations, inferences, or estimations. In further aspects, visual information can be utilized and/or stored in Extra Info **527**. Visual information can be useful in comparisons or decision making performed in autonomous Avatar **605** operation related to an object or environment that can be recognized from visual information. For example, an object or environment can be recognized by processing one or more digital pictures from Picture Renderer **91**, visual processor, visual program, or other visual provider. Any features, functionalities, and embodiments of Picture Recognizer **92** can be utilized for such recognizing as previously described. For instance, trees recognized in the background of one or more digital pictures from Picture Renderer **91** may indicate a park or forest. In further aspects, acoustic information can be utilized and/or stored in Extra Info **527**. Acoustic information can be useful in comparisons or decision making performed in autonomous Avatar **605** operation related to a sound or accosting environment. For example, an object or environment can be recognized by processing digital sound from Sound Renderer **96**, sound processor, sound program, or other sound provider. Any features, functionalities, and embodiments of Sound Recognizer **97** can be utilized for such recognizing as previously described. For instance, sound of a horn recognized in digital sound from Sound Renderer **96** may indicate

a proximal vehicle. In further aspects, other information can be utilized and/or stored in Extra Info **527**. Examples of such other information include user specific information (i.e. skill level, age, gender, etc.), group user information (i.e. access level, etc.), version of Application Program **18**, type of Application Program **18**, type of Avatar **605**, name of Avatar **605**, allegiance of Avatar **605**, type of Processor **11**, type of Computing Device **70**, and/or other information all of which can be obtained from various devices, systems, repositories, functions, or elements of Computing Device **70**, Processor **11**, Application Program **18**, Avatar **605**, and/or other processing elements.

Referring to FIG. **9**, an embodiment where ACAAO Unit **100** is part of or operating on Processor **11** is illustrated. In one example, ACAAO Unit **100** may be a hardware element or circuit embedded or built into Processor **11**. In another example, ACAAO Unit **100** may be a program operating on Processor **11**.

Referring to FIG. **10**, an embodiment where ACAAO Unit **100** resides on Server **96** accessible over Network **95** is illustrated. Any number of Computing Devices **70**, Processors **11**, Application Programs **18**, and/or other elements may connect to such remote ACAAO Unit **100** and the remote ACAAO Unit **100** may learn operations of their Avatars **605** in circumstances including objects with various properties. In turn, any number of Computing Devices **70**, Processors **11**, Application Programs **18**, and/or other elements can utilize the remote ACAAO Unit **100** for autonomous operation of their Avatars **605**. A remote ACAAO Unit **100** can be offered as a network service (i.e. online application, etc.). In some aspects, a remote ACAAO Unit **100** (i.e. global ACAAO Unit **100**, etc.) may reside on the Internet and be available to all the world's Computing Devices **70**, Processors **11**, Application Programs **18**, and/or other elements configured to transmit operations of their Avatars **605** in circumstances including objects with various properties and/or configured to utilize the remote ACAAO Unit **100** for autonomous operation of their Avatars **605**. For example, multiple players (i.e. Users **50**, etc.) may operate their Avatars **605** in a computer game (i.e. Application Program **18**, etc.) running on their respective Computing Devices **70** where the Computing Devices **70** and/or elements thereof may be configured to transmit Avatar's **605** operations in circumstances including objects with various properties to a remote ACAAO Unit **100**. Such remote ACAAO Unit **100** enables learning of the players' collective knowledge of operating Avatar **605** in circumstances including objects with various properties. Server **96** may be or include any type or form of a remote computing device such as an application server, a network service server, a cloud server, a cloud, and/or other remote computing device. Server **96** may include any features, functionalities, and embodiments of Computing Device **70**. It should be understood that Server **96** does not have to be a separate computing device and that Server **96**, its elements, or its functionalities can be implemented on Computing Device **70**. Network **95** may include various networks, connection types, protocols, interfaces, APIs, and/or other elements or techniques known in art all of which are within the scope of this disclosure. Any of the previously described networks, network or connection types, networking interfaces, and/or other networking elements or techniques can similarly be utilized. Any of the disclosed elements can reside on Server **96** in alternate implementations. In one example, Artificial Intelligence Unit **110** can reside on Server **96** and Acquisition Interface **120**, Modification Interface **130**, and/or Object Processing Unit **140** can reside on Computing Device **70**. In another example, Knowledgebase **530** (later described) can reside on Server **96** and the rest of the elements of ACAAO Unit **100** can reside on Computing Device **70**. Any other combination of local and remote elements can be implemented.

Referring to FIG. **11**, an embodiment of Artificial Intelligence Unit **110** is illustrated. Artificial Intelligence Unit **110** comprises interconnected Knowledge Structuring Unit **520**, Knowledgebase **530**, Decision-making Unit **540**, and Confirmation Unit **550**. Other additional elements can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate embodiments.

Artificial Intelligence Unit **110** comprises the functionality for learning Avatar's **605** operation in circumstances including objects with various properties. Artificial Intelligence Unit **110** comprises the functionality for learning one or more collections of object representations correlated with any instruction sets, data, and/or other information. In some aspects, Artificial Intelligence Unit **110** comprises the functionality for learning one or more Collections of Object Representations **525** representing Objects **615** in Avatar's **605** surrounding correlated with any Instruction Sets **526** and/or Extra Info **527**. The Instruction Sets **526** may be used or executed in operating Avatar **605**. In other aspects, Artificial Intelligence Unit **110** comprises the functionality for learning one or more Collections of Object Representations **525** representing Objects **615** in Avatar's **605** surrounding some of which may not be correlated with any Instruction Sets **526** and/or Extra Info **527**. Further, Artificial Intelligence Unit **110** comprises the functionality for anticipating Avatar's **605** operation in circumstances including objects with various properties. Artificial Intelligence Unit **110** comprises the functionality for anticipating one or more instruction sets, data, and/or other information. Artificial Intelligence Unit **110** comprises the functionality for anticipating one or more Instruction Sets **526** (i.e. anticipatory Instruction Sets **526**, etc.) based on one or more incoming Collections of Object Representations **525** representing Objects **615** in Avatar's **605** surrounding. The one or more Instruction Sets **526** (i.e. anticipatory Instruction Sets **526**, etc.) may be used or executed in Avatar's **605** autonomous operation. In some embodiments of Application Programs **18** that do not comprise Avatar **605** or rely on Avatar **605** for their operation, Artificial Intelligence Unit **110** comprises the functionality for learning Application Program's **18** operation in circumstances including objects with various properties similar to the learning functionalities described with respect to Avatar **605**. Also, in such embodiments, Artificial Intelligence Unit **110** comprises the functionality for anticipating Application Program's **18** operation in circumstances including objects with various properties similar to the anticipating functionalities described with respect to Avatar **605**. Artificial Intelligence Unit **110** also comprises other disclosed functionalities.

Knowledge Structuring Unit **520**, Knowledgebase **530**, and Decision-making Unit **540** are described later.

Confirmation Unit **550** comprises the functionality for confirming, modifying, evaluating (i.e. rating, etc.), and/or canceling one or more anticipatory Instruction Sets **526**, and/or other functionalities. Confirmation Unit **550** is an optional element that can be omitted depending on implementation. In some embodiments, Confirmation Unit **550** can serve as a means of confirming anticipatory Instruction Sets **526**. For example, Decision-making Unit **540** may determine one or more anticipatory Instruction Sets **526** and provide them to User **50** for confirmation. User **50** may be

provided with an interface (i.e. graphical user interface, selectable list of anticipatory Instruction Sets **526**, etc.) to approve or confirm execution of the anticipatory Instruction Sets **526**. In some aspects, Confirmation Unit **550** can automate User **50** confirmation. In one example, if one or more incoming Collections of Object Representations **525** from Object Processing Unit **140** and one or more Collections of Object Representations **525** from a Knowledge Cell **800** were found to be a perfect or highly similar match, anticipatory Instruction Sets **526** correlated with the one or more Collections of Object Representations **525** from the Knowledge Cell **800** can be automatically executed without User's **50** confirmation. Conversely, if one or more incoming Collections of Object Representations **525** from Object Processing Unit **140** and one or more Collections of Object Representations **525** from a Knowledge Cell **800** were found to be less than a highly similar match, anticipatory Instruction Sets **526** correlated with the one or more Collections of Object Representations **525** from the Knowledge Cell **800** can be presented to User **50** for confirmation and/or modifying. Any features, functionalities, and/or embodiments of Similarity Comparison **125** (later described) can be utilized for such similarity determination. In other embodiments, Confirmation Unit **550** can serve as a means of modifying or editing anticipatory Instruction Sets **526**. For example, Decision-making Unit **540** may determine one or more anticipatory Instruction Sets **526** and provide them to User **50** for modification. User **50** may be provided with an interface (i.e. graphical user interface, etc.) to modify the anticipatory Instruction Sets **526** before their execution. In further embodiments, Confirmation Unit **550** can serve as a means of evaluating or rating anticipatory Instruction Sets **526** if they matched User's **50** intended operation. For example, Decision-making Unit **540** may determine one or more anticipatory Instruction Sets **526**, which the system may automatically execute. User **50** may be provided with an interface (i.e. graphical user interface, etc.) to rate (i.e. on a scale from 0 to 1, etc.) how well Decision-making Unit **540** predicted the executed anticipatory Instruction Sets **526**. In some aspects, rating can be automatic and based on a particular function or method that rates how well the anticipatory Instruction Sets **526** matched the desired operation. In one example, a rating function or method can assign a higher rating to anticipatory Instruction Sets **526** that were least modified in the confirmation process. In another example, a rating function or method can assign a higher rating to anticipatory Instruction Sets **526** that were canceled least number of times by User **50**. Any other automatic rating function or method can be utilized. In yet other embodiments, Confirmation Unit **550** can serve as a means of canceling anticipatory Instruction Sets **526** if they did not match User's **50** intended operation. For example, Decision-making Unit **540** may determine one or more anticipatory Instruction Sets **526**, which the system may automatically execute. The system may save the state of Computing Device **70**, Processor **11** (save its register values, etc.), Application Program **18** (i.e. save its variables, data structures, objects, location of its current instruction, etc.), Avatar **605**, and/or other processing elements before executing anticipatory Instruction Sets **526**. User **50** may be provided with an interface (i.e. graphical user interface, selectable list of prior executed anticipatory Instruction Sets **526**, etc.) to cancel one or more of the prior executed anticipatory Instruction Sets **526**, and restore Computing Device **70**, Processor **11**, Application Program **18**, Avatar **605**, and/or other processing elements to a prior state. In some aspects, Confirmation Unit **550** can optionally be disabled or omitted in order to provide an uninterrupted operation of Avatar **605**, Processor **11**, and/or Application Program **18**. For example, a form based application may be suitable for implementing the user confirmation step, whereas, a game application may be less suitable for implementing such interrupting step due to the real time nature of game application's operation.

Referring to FIG. **12**, an embodiment of Knowledge Structuring Unit **520** correlating individual Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527** is illustrated. Knowledge Structuring Unit **520** comprises the functionality for structuring the knowledge of Avatar's **605** operation in circumstances including objects with various properties, and/or other functionalities. Knowledge Structuring Unit **520** comprises the functionality for correlating one or more Collections of Object Representations **525** representing Objects **615** in Avatar's **605** surrounding with any Instruction Sets **526** and/or Extra Info **527**. The Instruction Sets **526** may be used or executed in operating Avatar **605**. Knowledge Structuring Unit **520** comprises the functionality for creating or generating Knowledge Cell **800** and storing one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** into the Knowledge Cell **800**. As such, Knowledge Cell **800** comprises the functionality for storing one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**. Knowledge Cell **800** includes knowledge (i.e. unit of knowledge, etc.) of how Avatar **605** operated in a circumstance including objects with various properties. Once created or generated, Knowledge Cells **800** can be used in/as neurons, nodes, vertices, or other elements in any of the data structures or arrangements (i.e. neural networks, graphs, sequences, etc.) used for storing the knowledge of Avatar's **605** operation in circumstances including objects with various properties, thereby facilitating learning functionalities herein. It should be noted that Extra Info **527** may be optionally used in some implementations to enable enhanced comparisons or decision making in autonomous Avatar **605** operation where applicable, and that Extra Info **527** can be omitted in alternate implementations.

In some embodiments, Knowledge Structuring Unit **520** receives one or more Collections of Object Representations **525** from Object Processing Unit **140**. Knowledge Structuring Unit **520** may also receive one or more Instruction Sets **526** from Acquisition Interface **120**. Knowledge Structuring Unit **520** may further receive any Extra Info **527**. Although, Extra Info **527** is not shown in this and/or other figures for clarity of illustration, it should be noted that any Collection of Object Representations **525**, Instruction Set **526**, and/or other element may include or be associated with Extra Info **527**. Knowledge Structuring Unit **520** may correlate one or more Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527**. Knowledge Structuring Unit **520** may then create Knowledge Cell **800** and store the one or more Collections of Object Representations **525** correlated with Instruction Sets **526** and/or Extra Info **527** into the Knowledge Cell **800**. Knowledge Cell **800** may include any data structure or arrangement that can facilitate such storing. For example, Knowledge Structuring Unit **520** may create Knowledge Cell **800***ax* and structure within it Collection of Object Representations **525***a***1** correlated with Instruction Sets **526***a***1**-**526***a***3** and/or any Extra Info **527** (not shown). Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800***ax* a Collection of Object Representations **525***a***2** correlated with Instruction Set **526***a***4** and/or any Extra Info **527** (not shown). Knowl-

edge Structuring Unit **520** may further structure within Knowledge Cell **800***ax* a Collection of Object Representations **525***a***3** without a correlated Instruction Set **526** and/or Extra Info **527**. Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800***ax* a Collection of Object Representations **525***a***4** correlated with Instruction Sets **526***a***5**-**526***a***6** and/or any Extra Info **527** (not shown). Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800***ax* a Collection of Object Representations **525***a***5** without a correlated Instruction Set **526** and/or Extra Info **527**. Knowledge Structuring Unit **520** may structure within Knowledge Cell **800***ax* additional Collections of Object Representations **525** correlated with any number (including zero [i.e. uncorrelated]) of Instruction Sets **526** and/or Extra Info **527** by following similar logic as described above.

In some embodiments, Knowledge Structuring Unit **520** may correlate a Collection of Object Representations **525** with one or more temporally corresponding Instruction Sets **526** and/or Extra Info **527**. This way, Knowledge Structuring Unit **520** can structure the knowledge of Avatar's **605** operation at or around the time of generating Collections of Object Representations **525**. Such functionality enables spontaneous or seamless learning of Avatar's **605** operation in circumstances including objects with various properties as Avatar **605** is operated in real time. In some designs, Knowledge Structuring Unit **520** may receive a stream of Instruction Sets **526** used or executed to effect Avatar's **605** operations as well as a stream of Collections of Object Representations **525** representing Objects **615** in Avatar's **605** surrounding as the operations are performed. Knowledge Structuring Unit **520** can then correlate Collections of Object Representations **525** from the stream of Collections of Object Representations **525** with temporally corresponding Instruction Sets **526** from the stream of Instruction Sets **526** and/or any Extra Info **527**. Collections of Object Representations **525** without a temporally corresponding Instruction Set **526** may be uncorrelated, for instance. In some aspects, Instruction Sets **526** and/or Extra Info **527** that temporally correspond to a Collection of Object Representations **525** may include Instruction Sets **526** used and/or Extra Info **527** obtained at the time of generating the Collection of Object Representations **525**. In other aspects, Instruction Sets **526** and/or Extra Info **527** that temporally correspond to a Collection of Object Representations **525** may include Instruction Sets **526** used and/or Extra Info **527** obtained within a certain time period before and/or after generating the Collection of Object Representations **525**. For example, Instruction Sets **526** and/or Extra Info **527** that temporally correspond to a Collection of Object Representations **525** may include Instruction Sets **526** used and/or Extra Info **527** obtained within 50 milliseconds, 1 second, 3 seconds, 20 seconds, 1 minute, 41 minutes, 2 hours, or any other time period before and/or after generating the Collection of Object Representations **525**. Such time periods can be defined by a user, by ACAAO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. In other aspects, Instruction Sets **526** and/or Extra Info **527** that temporally correspond to a Collection of Object Representations **525** may include Instruction Sets **526** used and/or Extra Info **527** obtained from the time of generating the Collection of Object Representations **525** to the time of generating a next Collection of Object Representations **525**. In further aspects, Instruction Sets **526** and/or Extra Info **527** that temporally correspond to a Collection of Object Representations **525** may include Instruction Sets **526** used and/or Extra Info **527** obtained from the time of generating a previous Collection of Object Representations **525** to the time of generating the Collection of Object Representations **525**. Any other temporal relationship or correspondence between Collections of Object Representations **525** and correlated Instruction Sets **526** and/or Extra Info **527** can be implemented.

In some embodiments, Knowledge Structuring Unit **520** can structure the knowledge of Avatar's **605** operation in a circumstance including objects with various properties into any number of Knowledge Cells **800**. In some aspects, Knowledge Structuring Unit **520** can structure into a Knowledge Cell **800** a single Collection of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**. In other aspects, Knowledge Structuring Unit **520** can structure into a Knowledge Cell **800** any number (i.e. 2, 3, 6, 9, 21, 98, 3210, 13592, 513299, 9147224, etc.) of Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**. In a special case, Knowledge Structuring Unit **520** can structure all Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** into a single long Knowledge Cell **800**. In further aspects, Knowledge Structuring Unit **520** can structure Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** into a plurality of Knowledge Cells **800**. In a special case, Knowledge Structuring Unit **520** can store periodic streams of Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** into a plurality of Knowledge Cells **800** such as hourly, daily, weekly, monthly, yearly, or other periodic Knowledge Cells **800**.

Referring to FIG. **13**, another embodiment of Knowledge Structuring Unit **520** correlating individual Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527** is illustrated. In such embodiments, Knowledge Structuring Unit **520** may generate Knowledge Cells **800** each comprising a single Collection of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**.

Referring to FIG. **14**, an embodiment of Knowledge Structuring Unit **520** correlating streams of Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527** is illustrated. For example, Knowledge Structuring Unit **520** may create Knowledge Cell **800***ax* and structure within it a stream of Collections of Object Representations **525***a***1**-**525***an* correlated with Instruction Set **526***a***1** and/or any Extra Info **527** (not shown). Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800***ax* a stream of Collections of Object Representations **525***b***1**-**525***bn* correlated with Instruction Sets **526***a***2**-**526***a***4** and/or any Extra Info **527** (not shown). Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800***ax* a stream of Collections of Object Representations **525***c***1**-**525***cn* without correlated Instruction Sets **526** and/or Extra Info **527**. Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800***ax* a stream of Collections of Object Representations **525***d***1**-**525***dn* correlated with Instruction Sets **526***a***5**-**526***a***6** and/or any Extra Info **527** (not shown). Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800***ax* additional streams of Collections of Object Representations **525** correlated with any number (including zero [i.e. uncorrelated]) of Instruction Sets **526** and/or Extra Info **527** by following similar logic as described above. The number of Collections of Object Representations **525** in some or all streams of Collections of

Object Representations **525***a***1**-**525***an*, **525***b***1**-**525***bn*, **525***c***1**-**525***cn*, **525***d***1**-**525***dn*, etc. may be equal or different. It should be noted that n or other such letters or indicia may follow the sequence and/or context where they are indicated. Also, a same letter or indicia such as n may represent a different number in different sequences or elements of a drawing.

Referring to FIG. **15**, another embodiment of Knowledge Structuring Unit **520** correlating streams of Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527** is illustrated. In such embodiments, Knowledge Structuring Unit **520** may generate Knowledge Cells **800** each comprising a single stream of Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**.

Knowledgebase **530** comprises the functionality for storing knowledge of Avatar's **605** operation in circumstances including objects with various properties, and/or other functionalities. Knowledgebase **530** comprises the functionality for storing one or more Collections of Object Representations **525** representing Objects **615** in Avatar's **605** surrounding correlated with any Instruction Sets **526** and/or Extra Info **527**. The Instruction Sets **526** may be used or executed in operating Avatar **605**. Knowledgebase **530** comprises the functionality for storing one or more Knowledge Cells **800** each including one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**. In some aspects, Collections of Object Representations **525** correlated with Instruction Sets **526** and/or Extra Info **527** can be stored directly within Knowledgebase **530** without using Knowledge Cells **800** as the intermediary data structures. In some embodiments, Knowledgebase **530** may be or include Neural Network **530***a* (later described). In other embodiments, Knowledgebase **530** may be or include Graph **530***b* (later described). In further embodiments, Knowledgebase **530** may be or include Collection of Sequences **530***c* (later described). In further embodiments, Knowledgebase **530** may be or include Sequence **533** (later described). In further embodiments, Knowledgebase **530** may be or include Collection of Knowledge Cells **530***d* (later described). In general, Knowledgebase **530** may be or include any data structure or arrangement capable of storing knowledge of Avatar's **605** operation in circumstances including objects with various properties. Knowledgebase **530** may reside locally on Computing Device **70**, or remotely (i.e. remote Knowledgebase **530**, etc.) on a remote computing device (i.e. server, cloud, etc.) accessible over a network or an interface.

In some embodiments, Knowledgebase **530** from one Computing Device **70** or ACAAO Unit **100** can be transferred to one or more other Computing Devices **70** or ACAAO Units **100**. Therefore, the knowledge of Avatar's **605** operation in circumstances including objects with various properties learned and/or stored on one Computing Device **70** or ACAAO Unit **100** can be transferred to one or more other Computing Devices **70** or ACAAO Units **100**. In one example, Knowledgebase **530** can be copied or downloaded to a file or other repository from one Computing Device **70** or ACAAO Unit **100** and loaded or inserted into another Computing Device **70** or ACAAO Unit **100**. In another example, Knowledgebase **530** from one Computing Device **70** or ACAAO Unit **100** can be available on a server accessible by other Computing Devices **70** or ACAAO Units **100** over a network or an interface. Once loaded into or accessed by a receiving Computing Device **70** or ACAAO Unit **100**, the receiving Computing Device **70** or ACAAO Unit **100** can then implement the knowledge of Avatar's **605** operation in circumstances including objects with various properties learned or stored on the originating Computing Device **70** or ACAAO Unit **100**.

In some embodiments, multiple Knowledgebases **530** (i.e. Knowledgebases **530** from different Computing Devices **70** or ACAAO Units **100**, etc.) can be combined to accumulate collective knowledge of operating Avatar **605** in circumstances including objects with various properties. In one example, one Knowledgebase **530** can be appended to another Knowledgebase **530** such as appending one Collection of Sequences **530***c* (later described) to another Collection of Sequences **530***c*, appending one Sequence **533** (later described) to another Sequence **533**, appending one Collection of Knowledge Cells **530***d* (later described) to another Collection of Knowledge Cells **530***d*, and/or appending other data structures or elements thereof. In another example, one Knowledgebase **530** can be copied into another Knowledgebase **530** such as copying one Collection of Sequences **530***c* into another Collection of Sequences **530***c*, copying one Collection of Knowledge Cells **530***d* into another Collection of Knowledge Cells **530***d*, and/or copying other data structures or elements thereof. In a further example, in the case of Knowledgebase **530** being or including Graph **530***b* or graph-like data structure (i.e. Neural Network **530***a*, tree, etc.), a union can be utilized to combine two or more Graphs **530***b* or graph-like data structures. For instance, a union of two Graphs **530***b* or graph-like data structures may include a union of their vertex (i.e. node, etc.) sets and their edge (i.e. connection, etc.) sets. Any other operations or combination thereof on graphs or graph-like data structures can be utilized to combine Graphs **530***b* or graph-like data structures. In a further example, one Knowledgebase **530** can be combined with another Knowledgebase **530** through later described learning processes where Knowledge Cells **800** may be applied one at a time and connected with prior and/or subsequent Knowledge Cells **800** such as in Graph **530***b* or Neural Network **530***a*. In such embodiments, instead of Knowledge Cells **800** generated by Knowledge Structuring Unit **520**, the learning process may utilize Knowledge Cells **800** from one Knowledgebase **530** to apply them onto another Knowledgebase **530**. Any other techniques known in art including custom techniques for combining data structures can be utilized for combining Knowledgebases **530** in alternate implementations. In any of the aforementioned and/or other combining techniques, similarity of elements (i.e. nodes/vertices, edges/connections, etc.) can be utilized in determining whether an element from one Knowledgebase **530** matches an element from another Knowledgebase **530**, and substantially or otherwise similar elements may be considered a match for combining purposes in some designs. Any features, functionalities, and embodiments of Similarity Comparison **125** (later described) can be used in such similarity determinations. A combined Knowledgebase **530** can be offered as a network service (i.e. online application, etc.), downloadable file, or other repository to all ACAAO Units **100** configured to utilize the combined Knowledgebase **530**. For example, a computer game (i.e. Application Program **18**, etc.) including or interfaced with ACAAO Unit **100** having access to a combined Knowledgebase **530** can use a collective knowledge for Avatar's **605** operation in circumstances including objects with various properties learned from multiple players (i.e. Users **50**, etc.) for Avatar's **605** autonomous operation.

Referring to FIG. **16**, the disclosed artificially intelligent devices, systems, and methods for learning and/or using an avatar's circumstances for autonomous avatar operation

may include various artificial intelligence models and/or techniques. The disclosed devices, systems, and methods are independent of the artificial intelligence model and/or technique used and any model and/or technique can be utilized to facilitate the functionalities described herein. Examples of these models and/or techniques include deep learning, supervised learning, unsupervised learning, neural networks (i.e. convolutional neural network, recurrent neural network, deep neural network, etc.), search-based, logic and/or fuzzy logic-based, optimization-based, tree/graph/other data structure-based, hierarchical, symbolic and/or sub-symbolic, evolutionary, genetic, multi-agent, deterministic, probabilistic, statistical, and/or other models and/or techniques.

In one example shown in Model A, the disclosed artificially intelligent devices, systems, and methods for learning and/or using an avatar's circumstances for autonomous avatar operation may include a neural network (also referred to as artificial neural network, etc.). As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include a network of Nodes **852** (also referred to as neurons, etc.) and Connections **853** similar to that of a brain. Node **852** can store any data, object, data structure, and/or other item, or reference thereto. Node **852** may also include a function for transforming or manipulating any data, object, data structure, and/or other item. Examples of such transformation functions include mathematical functions (i.e. addition, subtraction, multiplication, division, sin, cos, log, derivative, integral, etc.), object manipulation functions (i.e. creating an object, modifying an object, deleting an object, appending objects, etc.), data structure manipulation functions (i.e. creating a data structure, modifying a data structure, deleting a data structure, creating a data field, modifying a data field, deleting a data field, etc.), and/or other transformation functions. Connection **853** may include or be associated with a value such as a symbolic label or numeric attribute (i.e. weight, cost, capacity, length, etc.). A computational model can be utilized to compute values from inputs based on a pre-programmed or learned function or method. For example, a neural network may include one or more input neurons that can be activated by inputs. Activations of these neurons can then be passed on, weighted, and transformed by a function to other neurons. Neural networks may range from those with only one layer of single direction logic to multi-layer of multi-directional feedback loops. A neural network can use weights to change the parameters of the network's throughput. A neural network can learn by input from its environment or from self-teaching using written-in rules. A neural network can be utilized as a predictive modeling approach in machine learning. An exemplary embodiment of a neural network (i.e. Neural Network **530***a*, etc.) is described later.

In another example shown in Model B, the disclosed artificially intelligent devices, systems, and methods for learning and/or using an avatar's circumstances for autonomous avatar operation may include a graph or graph-like data structure. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include Nodes **852** (also referred to as vertices or points, etc.) and Connections **853** (also referred to as edges, arrows, lines, arcs, etc.) organized as a graph. In general, any Node **852** in a graph can be connected to any other Node **852**. A Connection **853** may include unordered pair of Nodes **852** in an undirected graph or ordered pair of Nodes **852** in a directed graph. Nodes **852** can be part of the graph structure or external entities represented by indices or references. A graph can be utilized as a predictive modeling approach in machine learning. Nodes **852**, Connections **853**, and/or other elements or operations of a graph may include any features, functionalities, and embodiments of the aforementioned Nodes **852**, Connections **853**, and/or other elements or operations of a neural network, and vice versa. An exemplary embodiment of a graph (i.e. Graph **530***b*, etc.) is described later.

In a further example shown in Model C, the disclosed artificially intelligent devices, systems, and methods for learning and/or using an avatar's circumstances for autonomous avatar operation may include a tree or tree-like data structure. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include Nodes **852** and Connections **853** (also referred to as references, edges, etc.) organized as a tree. In general, a Node **852** in a tree can be connected to any number (i.e. including zero, etc.) of children Nodes **852**. A tree can be utilized as a predictive modeling approach in machine learning. Nodes **852**, Connections **853**, and/or other elements or operations of a tree may include any features, functionalities, and embodiments of the aforementioned Nodes **852**, Connections **853**, and/or other elements or operations of a neural network and/or graph, and vice versa.

In a further example shown in Model D, the disclosed artificially intelligent devices, systems, and methods for learning and/or using an avatar's circumstances for autonomous avatar operation may include a sequence or sequence-like data structure. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include a structure of Nodes **852** and/or Connections **853** organized as a sequence. In some aspects, Connections **853** may be optionally omitted from a sequence as the sequential order of Nodes **852** in a sequence may be implied in the structure. A sequence can be utilized as a predictive modeling approach in machine learning. Nodes **852**, Connections **853**, and/or other elements or operations of a sequence may include any features, functionalities, and embodiments of the aforementioned Nodes **852**, Connections **853**, and/or other elements or operations of a neural network, graph, and/or tree, and vice versa. An exemplary embodiment of a sequence (i.e. Collection of Sequences **530***c*, Sequence **533**, etc.) is described later.

In yet another example, the disclosed artificially intelligent devices, systems, and methods for learning and/or using an avatar's circumstances for autonomous avatar operation may include a search-based model and/or technique. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include searching through a collection of possible solutions. For example, a search method can search through a neural network, graph, tree, sequence, or other data structure that includes data elements of interest. A search may use heuristics to limit the search for solutions by eliminating choices that are unlikely to lead to the goal. Heuristic techniques may provide a best guess solution. A search can also include optimization. For example, a search may begin with a guess and then refine the guess incrementally until no more refinements can be made. In a further example, the disclosed systems, devices, and methods may include logic-based model and/or technique. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities can use formal or other type of logic. Logic based models may involve making

inferences or deriving conclusions from a set of premises. As such, a logic based system can extend existing knowledge or create new knowledge automatically using inferences. Examples of the types of logic that can be utilized include propositional or sentential logic that comprises logic of statements which can be true or false; first-order logic that allows the use of quantifiers and predicates and that can express facts about objects, their properties, and their relations with each other; fuzzy logic that allows degrees of truth to be represented as a value between 0 and 1 rather than simply 0 (false) or 1 (true), which can be used for uncertain reasoning; subjective logic that comprises a type of probabilistic logic that may take uncertainty and belief into account, which can be suitable for modeling and analyzing situations involving uncertainty, incomplete knowledge, and different world views; and/or other types of logic. In a further example, the disclosed systems, devices, and methods may include a probabilistic model and/or technique. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities can be implemented to operate with incomplete or uncertain information where probabilities may affect outcomes. Bayesian network, among other models, is an example of a probabilistic tool used for purposes such as reasoning, learning, planning, perception, and/or others. One of ordinary skill in art will understand that the aforementioned artificial intelligence models and/or techniques are described merely as examples of a variety of possible implementations, and that while all possible artificial intelligence models and/or techniques are too voluminous to describe, other artificial intelligence models and/or techniques known in art are within the scope of this disclosure. One of ordinary skill in art will also recognize that an intelligent system may solve a specific problem by using any model and/or technique that works such as, for example, some systems can be symbolic and logical, some can be sub-symbolic neural networks, some can be deterministic or probabilistic, some can be hierarchical, some may include searching techniques, some may include optimization techniques, while others may use other or a combination of models and/or techniques. In general, any artificial intelligence model and/or technique can be utilized that can facilitate the functionalities described herein.

Referring to FIGS. **17**A-**17**C, embodiments of interconnected Knowledge Cells **800** and updating weights of Connections **853** are illustrated. As shown for example in FIG. **17**A, Knowledge Cell **800***za* is connected to Knowledge Cell **800***zb* and Knowledge Cell **800***zc* by Connection **853***z***1** and Connection **853***z***2**, respectively. Each of Connection **853***z***1** and Connection **853***z***2** may include or be associated with occurrence count, weight, and/or other parameters or data. The number of occurrences may track or store the number of observations that a Knowledge Cell **800** was followed by another Knowledge Cell **800** indicating a connection or relationship between them. For example, Knowledge Cell **800***za* was followed by Knowledge Cell **800***zb* 10 times as indicated by the number of occurrences of Connection **853***z***1**. Also, Knowledge Cell **800***za* was followed by Knowledge Cell **800***zc* times as indicated by the number of occurrences of Connection **853***z***2**. The weight of Connection **853***z***1** can be calculated or determined as the number of occurrences of Connection **853***z***1** divided by the sum of occurrences of all connections (i.e. Connection **853***z***1** and Connection **853***z***2**, etc.) originating from Knowledge Cell **800***za*. Therefore, the weight of Connection **853***z***1** can be calculated or determined as 10/(10+15)=0.4, for example.

Also, the weight of Connection **853***z***2** can be calculated or determined as 15/(10+15)=0.6, for example. Therefore, the sum of weights of Connection **853***z***1**, Connection **853***z***2**, and/or any other Connections **853** originating from Knowledge Cell **800***za* may equal to 1 or 100%. As shown for example in FIG. **17**B, in the case that Knowledge Cell **800***zd* is inserted and an observation is made that Knowledge Cell **800***zd* follows Knowledge Cell **800***za*, Connection **853***z***3** can be created between Knowledge Cell **800***za* and Knowledge Cell **800***zd*. The occurrence count of Connection **853***z***3** can be set to 1 and weight determined as 1/(10+15+1)=0.038. The weights of all other connections (i.e. Connection **853***z***1**, Connection **853***z***2**, etc.) originating from Knowledge Cell **800***za* may be updated to account for the creation of Connection **853***z***3**. Therefore, the weight of Connection **853***z***1** can be updated as 10/(10+15+1)=0.385. The weight of Connection **853***z***2** can also be updated as 15/(10+15+1)=0.577. As shown for example in FIG. **17**C, in the case that an additional occurrence of Connection **853***z***1** is observed (i.e. Knowledge Cell **800***zb* followed Knowledge Cell **800***za*, etc.), occurrence count of Connection **853***z***1** and weights of all connections (i.e. Connection **853***z***1**, Connection **853***z***2**, and Connection **853***z***3**, etc.) originating from Knowledge Cell **800***za* may be updated to account for this observation. The occurrence count of Connection **853***z***1** can be increased by 1 and its weight updated as 11/(11+15+1)=0.407. The weight of Connection **853***z***2** can also be updated as 15/(11+15+1)=0.556. The weight of Connection **853***z***3** can also be updated as 1/(11+15+1)=0.037.

Referring to FIG. **18**, an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Collection of Knowledge Cells **530***d* is illustrated. Collection of Knowledge Cells **530***d* comprises the functionality for storing any number of Knowledge Cells **800**. In some aspects, Knowledge Cells **800** may be stored into or applied onto Collection of Knowledge Cells **530***d* in a learning or training process. In effect, Collection of Knowledge Cells **530***d* may store Knowledge Cells **800** that can later be used to enable autonomous Avatar **605** operation. In some embodiments, Knowledge Structuring Unit **520** structures or generates Knowledge Cells **800** as previously described and the system applies them onto Collection of Knowledge Cells **530***d*, thereby implementing learning Avatar's **605** operation in circumstances including objects with various properties. The term apply or applying may refer to storing, copying, inserting, updating, or other similar action, therefore, these terms may be used interchangeably herein depending on context. The system can perform Similarity Comparisons **125** (later described) of a newly structured Knowledge Cell **800** from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. If a substantially similar Knowledge Cell **800** is not found in Collection of Knowledge Cells **530***d*, the system may insert (i.e. copy, store, etc.) the Knowledge Cell **800** from Knowledge Structuring Unit **520** into Collection of Knowledge Cells **530***d*, for example. On the other hand, if a substantially similar Knowledge Cell **800** is found in Collection of Knowledge Cells **530***d*, the system may optionally omit inserting the Knowledge Cell **800** from Knowledge Structuring Unit **520** as inserting a substantially similar Knowledge Cell **800** may not add much or any additional knowledge to the Collection of Knowledge Cells **530***d*, for example. Also, inserting a substantially similar Knowledge Cell **800** can optionally be omitted to save storage resources and limit the number of Knowledge Cells **800** that may later

need to be processed or compared. Any features, functionalities, and embodiments of Similarity Comparison **125**, importance index (later described), similarity index (later described), and/or other disclosed elements can be utilized to facilitate determination of substantial or other similarity and whether to insert a newly structured Knowledge Cell **800** into Collection of Knowledge Cells **530***d*.

For example, the system can perform Similarity Comparisons **125** (later described) of Knowledge Cell **800***ba* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. In the case that a substantially similar match is found between Knowledge Cell **800***ba* and any of the Knowledge Cells **800** in Collection of Knowledge Cells **530***d*, the system may perform no action. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bb* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. In the case that a substantially similar match is not found, the system may insert a new Knowledge Cell **800** into Collection of Knowledge Cells **530***d* and copy Knowledge Cell **800***bb* into the inserted new Knowledge Cell **800**. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bc* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. In the case that a substantially similar match is found between Knowledge Cell **800***bc* and any of the Knowledge Cells **800** in Collection of Knowledge Cells **530***d*, the system may perform no action. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bd* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. In the case that a substantially similar match is not found, the system may insert a new Knowledge Cell **800** into Collection of Knowledge Cells **530***d* and copy Knowledge Cell **800***bd* into the inserted new Knowledge Cell **800**. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***be* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. In the case that a substantially similar match is not found, the system may insert a new Knowledge Cell **800** into Collection of Knowledge Cells **530***d* and copy Knowledge Cell **800***be* into the inserted new Knowledge Cell **800**. Applying any additional Knowledge Cells **800** from Knowledge Structuring Unit **520** onto Collection of Knowledge Cells **530***d* follows similar logic or process as the above-described.

Referring to FIG. **19**, an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Neural Network **530***a* is illustrated. Neural Network **530***a* includes a number of neurons or Nodes **852** interconnected by Connections **853** as previously described. Knowledge Cells **800** are shown instead of Nodes **852** to simplify the illustration as Node **852** includes a Knowledge Cell **800**, for example. Therefore, Knowledge Cells **800** and Nodes **852** can be used interchangeably herein depending on context. It should be noted that Node **852** may include other elements and/or functionalities instead of or in addition to Knowledge Cell **800**. In some aspects, Knowledge Cells **800** may be stored into or applied onto Neural Network **530***a* individually or collectively in a learning or training process. In some designs, Neural Network **530***a* comprises a number of Layers **854** each of which may include one or more Knowledge Cells **800**. Knowledge Cells **800** in successive Layers **854** can be connected by Connections **853**. Connection **853** may include or be associated with occurrence count, weight, and/or other parameter or data as previously described. Neural Network **530***a* may include any number of Layers **854** comprising any number of Knowledge Cells **800**. In some aspects, Neural Network **530***a* may store Knowledge Cells **800** interconnected by Connections **853** where following a path through the Neural Network **530***a* can later be used to enable autonomous Avatar **605** operation. It should be understood that, in some embodiments, Knowledge Cells **800** in one Layer **854** of Neural Network **530***a* need not be connected only with Knowledge Cells **800** in a successive Layer **854**, but also in any other Layer **854**, thereby creating shortcuts (i.e. shortcut Connections **853**, etc.) through Neural Network **530***a*. A Knowledge Cell **800** can also be connected to itself such as, for example, in recurrent neural networks. In general, any Knowledge Cell **800** can be connected with any other Knowledge Cell **800** anywhere else in Neural Network **530***a*. In further embodiments, back-propagation of any data or information can be implemented. In one example, back-propagation of similarity (i.e. similarity index, etc.) of compared Knowledge Cells **800** in a path through Neural Network **530***a* can be implemented. In another example, back-propagation of errors can be implemented. Such back-propagations can then be used to adjust occurrence counts and/or weights of Connections **853** for better future predictions, for example. Any other back-propagation can be implemented for other purposes. Any combination of Nodes **852** (i.e. Nodes **852** comprising Knowledge Cells **800**, etc.), Connections **853**, Layers **854**, and/or other elements or techniques can be implemented in alternate embodiments. Neural Network **530***a* may include any type or form of a neural network known in art such as a feed-forward neural network, a back-propagating neural network, a recurrent neural network, a convolutional neural network, deep neural network, and/or others including a custom neural network.

In some embodiments, Knowledge Structuring Unit **520** structures or generates Knowledge Cells **800** and the system applies them onto Neural Network **530***a*, thereby implementing learning Avatar's **605** operation in circumstances including objects with various properties. The system can perform Similarity Comparisons **125** (later described) of a Knowledge Cell **800** from Knowledge Structuring Unit **520** with Knowledge Cells **800** in a Layer **854** of Neural Network **530***a*. If a substantially similar Knowledge Cell **800** is not found in the Layer **854** of Neural Network **530***a*, the system may insert (i.e. copy, store, etc.) the Knowledge Cell **800** from Knowledge Structuring Unit **520** into the Layer **854** of Neural Network **530***a*, and create a Connection **853** to the inserted Knowledge Cell **800** from a Knowledge Cell **800** in a prior Layer **854** including assigning an occurrence count to the new Connection **853**, calculating a weight of the new Connection **853**, and updating any other Connections **853** originating from the Knowledge Cell **800** in the prior Layer **854**. On the other hand, if a substantially similar Knowledge Cell **800** is found in the Layer **854** of Neural Network **530***a*, the system may update occurrence count and weight of Connection **853** to that Knowledge Cell **800** from a Knowledge Cell **800** in a prior Layer **854**, and update any other Connections **853** originating from the Knowledge Cell **800** in the prior Layer **854**.

For example, the system can perform Similarity Comparisons **125** (later described) of Knowledge Cell **800***ba* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Layer **854***a* of Neural Network **530***a*. In the case that a substantially similar match is found between Knowledge Cell **800***ba* and Knowledge Cell **800***ea*, the system may perform no action since Knowledge Cell **800***ea* is the initial Knowledge Cell **800**. The system can then perform Simi-

larity Comparisons **125** of Knowledge Cell **800***bb* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Layer **854***b* of Neural Network **530***a*. In the case that a substantially similar match is found between Knowledge Cell **800***bb* and Knowledge Cell **800***eb*, the system may update occurrence count and weight of Connection **853***e***1** between Knowledge Cell **800***ea* and Knowledge Cell **800***eb*, and update weights of other Connections **853** originating from Knowledge Cell **800***ea* as previously described. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bc* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Layer **854***c* of Neural Network **530***a*. In the case that a substantially similar match is not found, the system may insert Knowledge Cell **800***ec* into Layer **854***c* and copy Knowledge Cell **800***bc* into the inserted Knowledge Cell **800***ec*. The system may also create Connection **853***e***2** between Knowledge Cell **800***eb* and Knowledge Cell **800***ec* with occurrence count of 1 and weight calculated based on the occurrence count as previously described. The system may also update weights of other Connections **853** (one in this example) originating from Knowledge Cell **800***eb* as previously described. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bd* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Layer **854***d* of Neural Network **530***a*. In the case that a substantially similar match is not found, the system may insert Knowledge Cell **800***ed* into Layer **854***d* and copy Knowledge Cell **800***bd* into the inserted Knowledge Cell **800***ed*. The system may also create Connection **853***e***3** between Knowledge Cell **800***ec* and Knowledge Cell **800***ed* with occurrence count of 1 and weight of 1. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***be* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Layer **854***e* of Neural Network **530***a*. In the case that a substantially similar match is not found, the system may insert Knowledge Cell **800***ee* into Layer **854***e* and copy Knowledge Cell **800***be* into the inserted Knowledge Cell **800***ee*. The system may also create Connection **853***e***4** between Knowledge Cell **800***ed* and Knowledge Cell **800***ee* with occurrence count of 1 and weight of 1. Applying any additional Knowledge Cells **800** from Knowledge Structuring Unit **520** onto Neural Network **530***a* follows similar logic or process as the above-described.

Referring now to Similarity Comparison **125**, Similarity Comparison **125** comprises the functionality for comparing or matching Knowledge Cells **800** or portions thereof, and/or other functionalities. Similarity Comparison **125** comprises the functionality for comparing or matching Collections of Object Representations **525** or portions thereof. Similarity Comparison **125** comprises the functionality for comparing or matching streams of Collections of Object Representations **525** or portions thereof. Similarity Comparison **125** comprises the functionality for comparing or matching Object Representations **625** or portions thereof. Similarity Comparison **125** comprises the functionality for comparing or matching Object Properties **630** or portions thereof. Similarity Comparison **125** comprises the functionality for comparing or matching Instruction Sets **526**, Extra Info **527**, text (i.e. characters, words, phrases, etc.), numbers, and/or other elements or portions thereof. Similarity Comparison **125** may include functions, rules, and/or logic for performing matching or comparisons and for determining that while a perfect match is not found, a partial or similar match has been found. In some aspects, a partial match may include a substantially or otherwise similar match, and vice versa. Therefore, these terms may be used interchangeably herein depending on context. As such, Similarity Comparison **125** may include determining substantial similarity or substantial match of compared elements. Although, substantial similarity or substantial match is frequently used herein, it should be understood that any level of similarity, however high or low, may be utilized as defined by the rules (i.e. thresholds, etc.) for similarity. The rules for similarity or similar match can be defined by a user, by ACAAO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. In some designs, Similarity Comparison **125** comprises the functionality to automatically define appropriately strict rules for determining similarity of the compared elements. Similarity Comparison **125** can therefore set, reset, and/or adjust the strictness of the rules for finding or determining similarity of the compared elements, thereby fine tuning Similarity Comparison **125** so that the rules for determining similarity are appropriately strict. In some aspects, the rules for determining similarity may include a similarity threshold. As such, Similarity Comparison **125** can determine similarity of compared elements if their similarity exceeds a similarity threshold. In other aspects, the rules for determining similarity may include a difference threshold. As such, Similarity Comparison **125** can determine similarity of compared elements if their difference is lower than a difference threshold. In further aspects, the rules for determining similarity may include other thresholds. Similarity Comparison **125** enables comparing circumstances including objects with various properties and determining their similarity or match. In one example, a circumstance including an object located at a distance of 9 m and an angle/bearing of 97° relative to Avatar **605** may be found similar or matching by Similarity Comparison **125** to a circumstance including the same or similar object located at a distance of 8.7 m and an angle/bearing of 101° relative to Avatar **605**. In another example, a circumstance including an object detected as a female person may be found similar or matching by Similarity Comparison **125** to a circumstance including an object detected as a male person. In general, any one or more properties (i.e. existence, type, identity, distance, bearing/angle, location, shape/size, activity, etc.) of one or more objects can be utilized for determining similarity or match of circumstances including objects with various properties. Therefore, Similarity Comparison **125** provides flexibility in comparing and determining similarity of a variety of possible circumstances of Avatar **605**.

In some embodiments where compared Knowledge Cells **800** include a single Collection of Object Representations **525**, in determining similarity of Knowledge Cells **800**, Similarity Comparison **125** can perform comparison of individual Collections of Object Representations **525** or portions (i.e. Object Representations **625**, Object Properties **630**, etc.) thereof such as comparison of Collection of Object Representations **525** or portions thereof from one Knowledge Cell **800** with Collection of Object Representations **525** or portions thereof from another Knowledge Cell **800**. In some aspects, total equivalence is achieved when Collection of Object Representations **525** or portions thereof from one Knowledge Cell **800** matches Collection of Object Representations **525** or portions thereof from another Knowledge Cell **800**. If total equivalence is not found, Similarity Comparison **125** may attempt to determine substantial or other similarity of compared Knowledge Cells **800**.

In some embodiments, in determining substantial similarity of individually compared Collections of Object Representations **525** (i.e. Collections of Object Representations

525 from the compared Knowledge Cells **800**, etc.), Similarity Comparison **125** can compare one or more Object Representations **625** or portions (i.e. Object Properties **630**, etc.) thereof from one Collection of Object Representations **525** with one or more Object Representations **625** or portions thereof from another Collection of Object Representations **525**. In some aspects, total equivalence is found when all Object Representations **625** or portions thereof from one Collection of Object Representations **525** match all Object Representations **625** or portions thereof from another Collection of Object Representations **525**. In other aspects, if total equivalence is not found, Similarity Comparison **125** may attempt to determine substantial similarity of compared Collections of Object Representations **525**. In one example, substantial similarity can be achieved when most of the Object Representations **625** or portions thereof from the compared Collections of Object Representations **525** match or substantially match. In another example, substantial similarity can be achieved when at least a threshold number (i.e. 1, 2, 4, 7, 18, etc.) or percentage (i.e. 41%, 62%, 79%, 85%, 93%, etc.) of Object Representations **625** or portions thereof from the compared Collections of Object Representations **525** match or substantially match. Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching Object Representations **625** or portions thereof from the compared Collections of Object Representations **525** exceeds a threshold number (i.e. 1, 2, 4, 7, 18, etc.) or a threshold percentage (i.e. 41%, 62%, 79%, 85%, 93%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or percentage of Object Representations **625** or portions thereof from the compared Collections of Object Representations **525** match or substantially match. Such thresholds can be defined by a user, by ACAAO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In some aspects, Similarity Comparison **125** can utilize the importance (i.e. as indicated by importance index [ater described], etc.) of Object Representations **625** or portions thereof for determining substantial similarity of Collections of Object Representations **525**. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to more important Object Representations **625** or portions thereof such as Object Representations **625** representing near Objects **615**, Object Representations **625** representing large Objects **615**, etc., thereby tolerating mismatches in less important Object Representations **625** or portions thereof such as Object Representations **625** representing distant Objects **615**, Object Representations **625** representing small Objects **615**, etc. In general, any Object Representation **625** or portion thereof can be assigned higher or lower importance. In further aspects, Similarity Comparison **125** can omit some of the Object Representations **625** or portions thereof from the comparison in determining substantial similarity of Collections of Object Representations **525**. In one example, Object Representations **625** representing distant Objects **615** can be omitted from comparison. In another example, Object Representations **625** representing small Objects **615** can be omitted from comparison. In general, any Object Representation **625** or portion thereof can be omitted from comparison depending on implementation.

Similarity Comparison **125** can automatically adjust (i.e. increase or decrease) the strictness of the rules for determining substantial similarity of Collections of Object Representations **525**. In some aspects, such adjustment in strictness can be done by Similarity Comparison **125** in response to determining that total equivalence of compared Collections of Object Representations **525** had not been found. Similarity Comparison **125** can keep adjusting the strictness rules until a substantial similarity is found. All the rules or settings of substantial similarity can be set, reset, or adjusted by Similarity Comparison **125** in response to another strictness level determination. For example, Similarity Comparison **125** may attempt to find a match or substantial match in a certain percentage (i.e. 83%, etc.) of Object Representations **625** or portions thereof from the compared Collections of Object Representations **525**. If the comparison does not determine substantial similarity of compared Collections of Object Representations **525**, Similarity Comparison **125** may decide to decrease the strictness of the rules. In response, Similarity Comparison **125** may attempt to find fewer matching or substantially matching Object Representations **625** or portions thereof than in the previous attempt using stricter rules. If the comparison still does not determine substantial similarity of compared Collections of Object Representations **525**, Similarity Comparison **125** may further decrease the strictness (i.e. down to a certain minimum strictness or threshold, etc.) by requiring fewer Object Representations **625** or portions thereof to match or substantially match, thereby further increasing a chance of finding substantial similarity in compared Collections of Object Representations **525**.

Where a reference to Object Representation **625** is used herein it should be understood that a portion of Object Representation **625** (i.e. Object Property **630**, etc.) or a plurality of Object Representations **625** can be used instead of or in addition to the Object Representation **625**. In one example, instead of or in addition to Object Representation **625**, Object Properties **630** and/or other portions that constitute an Object Representation **625** can be compared. In another example, instead of or in addition to Object Representation **625**, plurality of Object Representations **625** can be compared. As such, any operations, rules, logic, and/or functions operating on Object Representation **625** may similarly apply to any portion of Object Representation **625** and/or a plurality of Object Representations **625** as applicable. In general, whole Object Representations **625**, portions of Object Representations **625**, and/or pluralities of Object Representations **625**, including any operations thereon, can be combined to arrive at desired results. Some or all of the above-described rules, logic, and/or techniques can be utilized alone or in combination with each other or with other rules, logic, and/or techniques. One of ordinary skill in art will recognize that other techniques known in art for determining similarity of Object Representations **625** and/or other data that would be too voluminous to describe are within the scope of this disclosure.

In some embodiments, in determining substantial similarity of Object Representations **625** (i.e. Object Representations **625** from the compared Collections of Object Representations **525**, etc.), Similarity Comparison **125** can compare Object Properties **630** or portions (i.e. characters, words, numbers, etc.) thereof from one Object Representation **625** with Object Properties **630** or portions thereof from another Object Representation **625**. In some aspects, total equivalence is found when all Object Properties **630** or portions thereof of one Object Representation **625** match all Object Properties **630** or portions thereof of another Object Representation **625**. In other aspects, if total equivalence is not found, Similarity Comparison **125** may attempt to determine substantial similarity of compared Object Representations **625**. In one example, substantial similarity can be achieved when most of the Object Properties **630** or portions

thereof from the compared Object Representations **625** match or substantially match. In another example, substantial similarity can be achieved when at least a threshold number (i.e. 1, 2, 3, 6, 11, etc.) or percentage (i.e. 55%, 61%, 78%, 82%, 99%, etc.) of Object Properties **630** or portions thereof from the compared Object Representations **625** match or substantially match. Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching Object Properties **630** or portions thereof from the compared Object Representations **625** exceeds a threshold number (i.e. 1, 2, 3, 6, 11, etc.) or a threshold percentage (i.e. 55%, 61%, 78%, 82%, 99%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or percentage of Object Properties **630** or portions thereof from the compared Object Representations **625** match or substantially match. Such thresholds can be defined by a user, by ACAAO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In further aspects, Similarity Comparison **125** can utilize Categories **635** associated with Object Properties **630** for determining substantial similarity of Object Representations **625**. In one example, Object Properties **630** or portions thereof from the compared Object Representations **625** in a same Category **635** may be compared. This way, Object Properties **630** or portions thereof can be compared with their own peers. In one instance, Object Properties **630** or portions thereof from the compared Object Representations **625** in Category **635** "Type" may be compared. Any text comparison technique can be utilized in such comparing. In another instance, Object Properties **630** or portions thereof from the compared Object Representations **625** in Category **635** "Distance" or "Bearing" may be compared. Any number comparison technique can be utilized in such comparing. In a further instance, Object Properties **630** or portions thereof from the compared Object Representations **625** in Category **635** "Shape" may be compared. Any model or other computer construct comparison technique can be utilized in such comparing. In further aspects, Similarity Comparison **125** can utilize the importance (i.e. as indicated by importance index [later described], etc.) of Object Properties **630** or portions thereof for determining substantial similarity of Object Representations **625**. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to more important Object Properties **630** or portions thereof such as Object Properties **630** or portions thereof in Categories **635** "Type", "Distance", "Bearing", etc., thereby tolerating mismatches in less important Object Properties **630** or portions thereof such as Object Properties **630** or portions thereof in Categories **635** "Identity", "Shape", etc. In general, any Object Property **630** or portion thereof can be assigned higher or lower importance. In further aspects, Similarity Comparison **125** can omit some of the Object Properties **630** or portions thereof from the comparison in determining substantial similarity of Object Representations **625**. In one example, Object Properties **630** or portions thereof in Category **635** "Identity" can be omitted from comparison. In another example, Object Properties **630** or portions thereof in Category **635** "Shape" can be omitted from comparison. In general, any Object Property **630** or portion thereof can be omitted from comparison depending on implementation.

Similarity Comparison **125** can automatically adjust (i.e. increase or decrease) the strictness of the rules for determining substantial similarity of Object Representations **625**. In some aspects, such adjustment in strictness can be done by Similarity Comparison **125** in response to determining that total equivalence of compared Object Representations **625** had not been found. Similarity Comparison **125** can keep adjusting the strictness rules until a substantial similarity is found. All the rules or settings of substantial similarity can be set, reset, or adjusted by Similarity Comparison **125** in response to another strictness level determination. For example, Similarity Comparison **125** may attempt to find a match or substantial match in a certain percentage (i.e. 90%, etc.) of Object Properties **630** or portions thereof from the compared Object Representations **625**. If the comparison does not determine substantial similarity of compared Object Representations **625**, Similarity Comparison **125** may decide to decrease the strictness of the rules. In response, Similarity Comparison **125** may attempt to find fewer matching or substantially matching Object Properties **630** or portions thereof than in the previous attempt using stricter rules. If the comparison still does not determine substantial similarity of compared Object Representations **625**, Similarity Comparison **125** may further decrease the strictness (i.e. down to a certain minimum strictness or threshold, etc.) by requiring fewer Object Properties **630** or portions thereof to match or substantially match, thereby further increasing a chance of finding substantial similarity in compared Object Representations **625**. In further aspects, an adjustment in strictness can be done by Similarity Comparison **125** in response to determining that multiple substantially similar Object Representations **625** had been found. Similarity Comparison **125** can keep adjusting the strictness of the rules until a best of the substantially similar Object Representations **625** is found. For example, Similarity Comparison **125** may attempt to find a match or substantial match in a certain percentage (i.e. 67%, etc.) of Object Properties **630** or portions thereof from the compared Object Representations **625**. If the comparison determines a number of substantially similar Object Representations **625**, Similarity Comparison **125** may decide to increase the strictness of the rules to decrease the number of substantially similar Object Representations **625**. In response, Similarity Comparison **125** may attempt to find more matching or substantially matching Object Properties **630** or portions thereof in addition to the earlier found Object Properties **630** or portions thereof to limit the number of substantially similar Object Representations **625**. If the comparison still provides more than one substantially similar Object Representation **625**, Similarity Comparison **125** may further increase the strictness by requiring additional Object Properties **630** or portions thereof to match or substantially match, thereby further narrowing the number of substantially similar Object Representations **625** until a best substantially similar Object Representation **625** is found.

Where a reference to Object Property **630** is used herein it should be understood that a portion of Object Property **630** or a plurality of Object Properties **630** can be used instead of or in addition to the Object Property **630**. In one example, instead of or in addition to Object Property **630**, characters, words, numbers, and/or other portions that constitute an Object Property **630** can be compared. In another example, instead of or in addition to Object Property **630**, a plurality of Object Properties **630** can be compared. As such, any operations, rules, logic, and/or functions operating on Object Property **630** may similarly apply to any portion of Object Property **630** and/or a plurality of Object Properties **630** as applicable. In general, whole Object Properties **630**, portions of Object Properties **630**, and/or pluralities of Object Properties **630**, including any operations thereon, can be combined to arrive at desired results. Some or all of the above-

described rules, logic, and/or techniques can be utilized alone or in combination with each other or with other rules, logic, and/or techniques. One of ordinary skill in art will recognize that other techniques known in art for determining similarity of Object Properties **630** and/or other data that would be too voluminous to describe are within the scope of this disclosure.

In some embodiments where compared Knowledge Cells **800** include a stream of Collections of Object Representations **525**, in determining similarity of Knowledge Cells **800**, Similarity Comparison **125** can perform collective comparison of Collections of Object Representations **525** or portions (i.e. Object Representations **625**, Object Properties **630**, etc.) thereof such as comparison of a stream of Collections of Object Representations **525** or portions thereof from one Knowledge Cell **800** with a stream of Collections of Object Representations **525** or portions thereof from another Knowledge Cell **800**. Similarity Comparison **125** of collectively compared Collections of Object Representations **525** or portions thereof may include any features, functionalities, and embodiments of the previously described Similarity Comparison **125** of individually compared Collections of Object Representations **525** or portions thereof. In some aspects, total equivalence is found when all Collections of Object Representations **525** or portions thereof from one Knowledge Cell **800** match all Collections of Object Representations **525** or portions thereof from another Knowledge Cell **800**. If total equivalence is not found, Similarity Comparison **125** may attempt to determine substantial or other similarity of compared Knowledge Cells **800**. In one example, substantial similarity can be achieved when most of the Collections of Object Representations **525** or portions thereof from the compared Knowledge Cells **800** match or substantially match. In another example, substantial similarity can be achieved when at least a threshold number (i.e. 1, 2, 4, 9, 33, 138, etc.) or percentage (i.e. 39%, 58%, 77%, 88%, 94%, etc.) of Collections of Object Representations **525** or portions thereof from the compared Knowledge Cells **800** match or substantially match. Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching Collections of Object Representations **525** or portions thereof from the compared Knowledge Cells **800** exceeds a threshold number (i.e. 1, 2, 4, 9, 33, 138, etc.) or a threshold percentage (i.e. 39%, 58%, 77%, 88%, 94%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or percentage of Collections of Object Representations **525** or portions thereof from the compared Knowledge Cells **800** match or substantially match. Such thresholds can be defined by a user, by ACAAO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In some aspects, Similarity Comparison **125** can utilize the importance (i.e. as indicated by importance index [later described], etc.) of Collections of Object Representations **525** or portions thereof for determining substantial similarity of Knowledge Cells **800**. In one example, substantial similarity can be achieved when matches or substantial matches are found with respect to more important Collections of Object Representations **525** or portions thereof such as more substantive or larger Collections of Object Representations **525** (i.e. Collections of Object Representations **525** comprising a higher number of Object Representations **625**, etc.) or portions thereof, etc., thereby tolerating mismatches in less important Collections of Object Representations **525** or portions thereof such as less substantive or smaller Collections of Object Representations **525** (i.e. Collections of Object Representations **525** comprising a lower number of Object Representations **625**, etc.) or portions thereof, etc. In general, any Collection of Object Representations **525** or portion thereof can be assigned higher or lower importance. In other aspects, Similarity Comparison **125** can utilize the order of Collections of Object Representations **525** or portions thereof for determining substantial similarity of Knowledge Cells **800**. In one example, substantial similarity can be achieved when matches or substantial matches are found in earlier Collections of Object Representations **525** or portions thereof from the compared Knowledge Cells **800**, thereby tolerating mismatches in later Collections of Object Representations **525** or portions thereof. In another example, substantial similarity can be achieved when matches or substantial matches are found in corresponding (i.e. similarly ordered, temporally related, etc.) Collections of Object Representations **525** or portions thereof from the compared Knowledge Cells **800**. In one instance, a 86^{th} Collection of Object Representations **525** or portions thereof from one Knowledge Cell **800** can be compared with a 86^{th} Collection of Object Representations **525** or portions thereof from another Knowledge Cell **800**. In another instance, a 86^{th} Collection of Object Representations **525** or portions thereof from one Knowledge Cell **800** can be compared with a number of Collections of Object Representations **525** or portions thereof around (i.e. preceding and/or following) a 86^{th} Collection of Object Representations **525** from another Knowledge Cell **800**. This way, flexibility can be implemented in finding a substantially similar Collection of Object Representations **525** or portions thereof if the Collections of Object Representations **525** or portions thereof in the compared Knowledge Cells **800** are not perfectly aligned. In a further instance, Similarity Comparison **125** can utilize Dynamic Time Warping (DTW) and/or other techniques known in art for comparing and/or aligning temporal sequences (i.e. streams of Collections of Object Representations **525** or portions thereof, etc.) that may vary in time or speed. In further aspects, Similarity Comparison **125** can omit some of the Collections of Object Representations **525** or portions thereof from the comparison in determining substantial similarity of Knowledge Cells **800**. In one example, less substantive or smaller Collections of Object Representations **525** or portions thereof can be omitted from comparison. In another example, some or all Collections of Object Representations **525** or portions thereof related to a specific time period can be omitted from comparison. In general, any Collection of Object Representations **525** or portion thereof can be omitted from comparison depending on implementation.

Similarity Comparison **125** can automatically adjust (i.e. increase or decrease) the strictness of the rules for determining substantial similarity of Knowledge Cells **800**. In some aspects, such adjustment in strictness can be done by Similarity Comparison **125** in response to determining that total equivalence of compared Knowledge Cells **800** had not been found. Similarity Comparison **125** can keep adjusting the strictness of the rules until substantial similarity is found. All the rules or settings of substantial similarity can be set, reset, or adjusted by Similarity Comparison **125** in response to another strictness level determination. For example, Similarity Comparison **125** may attempt to find a match or substantial match in a certain percentage (i.e. 89%, etc.) of Collections of Object Representations **525** or portions thereof from the compared Knowledge Cells **800**. If the comparison does not determine substantial similarity of compared Knowledge Cells **800**, Similarity Comparison **125**

may decide to decrease the strictness of the rules. In response, Similarity Comparison **125** may attempt to find fewer matching or substantially matching Collections of Object Representations **525** or portions thereof than in the previous attempt using stricter rules. If the comparison still does not determine substantial similarity of compared Knowledge Cells **800**, Similarity Comparison **125** may further decrease (i.e. down to a certain minimum strictness or threshold, etc.) the strictness by requiring fewer Collections of Object Representations **525** or portions thereof to match or substantially match, thereby further increasing a chance of finding substantial similarity in compared Knowledge Cells **800**. In further aspects, an adjustment in strictness can be done by Similarity Comparison **125** in response to determining that multiple substantially similar Knowledge Cells **800** had been found. Similarity Comparison **125** can keep adjusting the strictness of the rules until a best of the substantially similar Knowledge Cells **800** is found. For example, Similarity Comparison **125** may attempt to find a match or substantial match in a certain percentage (i.e. 69%, etc.) of Collections of Object Representations **525** or portions thereof from the compared Knowledge Cells **800**. If the comparison determines a number of substantially similar Knowledge Cells **800**, Similarity Comparison **125** may decide to increase the strictness of the rules to decrease the number of substantially similar Knowledge Cells **800**. In response, Similarity Comparison **125** may attempt to find more matching or substantially matching Collections of Object Representations **525** or portions thereof in addition to the earlier found Collections of Object Representations **525** or portions thereof to limit the number of substantially similar Knowledge Cells **800**. If the comparison still provides more than one substantially similar Knowledge Cell **800**, Similarity Comparison **125** may further increase the strictness by requiring additional Collections of Object Representations **525** or portions thereof to match or substantially match, thereby further narrowing the number of substantially similar Knowledge Cells **800** until a best substantially similar Knowledge Cell **800** is found.

Some or all of the aforementioned rules, logic, and/or techniques for determining substantial similarity of Knowledge Cells **800** can be utilized alone or in combination with each other or with other rules, logic, and/or techniques. One of ordinary skill in art will recognize that other techniques known in art for determining similarity of Knowledge Cells **800** and/or other data that would be too voluminous to describe are within the scope of this disclosure.

In any of the comparisons involving numbers such as, for example, Object Properties **630** including numbers (i.e. distances, bearings/angles, etc.), Similarity Comparison **125** can compare a number from one Object Property **630** with a number from another Object Property **630**. In some aspects, total equivalence is found when the number from one Object Property **630** equals the number from another Object Property **630**. In other aspects, if total equality is not found, Similarity Comparison **125** may attempt to determine substantial similarity of the compared numbers using a tolerance or threshold for determining a match. In some aspects, Similarity Comparison **125** can utilize a threshold for acceptable number difference in determining a match of compared numbers. For example, a threshold for acceptable number difference (i.e. absolute difference, etc.) can be set at 10. Therefore, 130 matches or is sufficiently similar to 135 because the number difference (i.e. 5 in this example) is lower than the threshold for acceptable number difference (i.e. 10 in this example, etc.). Furthermore, 130 does not match or is not sufficiently similar to 143 because the number difference (i.e. 13 in this example) is greater than the threshold for acceptable number difference. Any other threshold for acceptable number difference can be used such as 0.024, 1, 8, 15, 77, 197, 2438, 728322, and/or others. In other aspects, Similarity Comparison **125** can utilize a threshold for acceptable percentage difference in determining a match of compared numbers. For example, a threshold for acceptable percentage difference can be set at 10%. Therefore, 100 matches or is sufficiently similar to 106 because the percentage difference (i.e. 6% in this example) is lower than the threshold for acceptable percentage difference (i.e. 10% in this example). Furthermore, 100 does not match or is not sufficiently similar to 84 because the percentage difference (i.e. 16% in this example) is higher than the threshold for acceptable percentage difference. Any other threshold for acceptable percentage difference can be used such as 0.68%, 1%, 3%, 11%, 33%, 69%, 87%, and/or others. The aforementioned thresholds can be defined by a user, by ACAAO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. Other techniques known in art for comparing numbers can be utilized herein.

In any of the comparisons involving text such as, for example, Object Properties **630** including text (i.e. types, identities, etc.), Similarity Comparison **125** can compare words, characters, and/or other text from one Object Property **630** with words, characters, and/or other text from another Object Property **630**. In some aspects, total equivalence is found when all words, characters, and/or other text from one Object Property **630** match all words, characters, and/or other text from another Object Property **630**. In other aspects, if total equivalence is not found, Similarity Comparison **125** may attempt to determine substantial similarity of compared Object Properties **630**. In one example, substantial similarity can be achieved when most of the words, characters, and/or other text from the compared Object Properties **630** match or substantially match. In another example, substantial similarity can be achieved when at least a threshold number (i.e. 1, 2, 3, 4, 7, 11, etc.) or percentage (i.e. 38%, 63%, 77%, 84%, 98%, etc.) of words, characters, and/or other text from the compared Object Properties **630** match or substantially match. Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching words, characters, and/or other text from the compared Object Properties **630** exceeds a threshold number (i.e. 1, 2, 3, 4, 7, 11, etc.) or a threshold percentage (i.e. 48%, 63%, 77%, 84%, 98%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or percentage of words, characters, and/or other text from the compared Object Properties **630** match or substantially match. Such thresholds can be defined by a user, by ACAAO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In further aspects, Similarity Comparison **125** can utilize the importance (i.e. as indicated by importance index [later described], etc.) of words, characters, and/or other text for determining substantial similarity of Object Properties **630**. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to more important words, characters, and/or other text such as longer words and/or other text, thereby tolerating mismatches in less important words, characters, and/or other text such as shorter words and/or other text. In general, any word, character, and/or other text can be assigned higher or lower importance. In further

aspects, Similarity Comparison **125** can utilize the order of words, characters, and/or other text for determining substantial similarity of Object Properties **630**. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to front-most words, characters, and/or other text, thereby tolerating mismatches in later words, characters, and/or other text. In further aspects, Similarity Comparison **125** can utilize semantic conversion to account for variations of words and/or other text. In one example, Object Property **630** may include a word "house". In addition to searching for the exact word in a compared Object Property **630**, Similarity Comparison **125** can employ semantic conversion and attempt to match "home", "residence", "dwelling", "place", or other semantically similar variations of the word with a meaning "house". In another example, Object Property **630** may include a word "buy". In addition to searching for the exact word in a compared Object Property **630**, Similarity Comparison **125** can employ semantic conversion and attempt to match "buying", "bought", or other semantically similar variations of the word with a meaning "buy" in different tenses. Any other grammatical analysis or transformation can be utilized to cover the full scope of word and/or other text variations. In some designs, semantic conversion can be implemented using a thesaurus or dictionary. In another example, semantic conversion can be implemented using a table where each row comprises semantically similar variations of a word and/or other text. In further aspects, Similarity Comparison **125** can utilize a language model for understanding or interpreting the concepts contained in the words and/or other text and compare the concepts instead of or in addition to the words and/or other text. Examples of language models include unigram model, n-gram model, neural network language model, bag of words model, and/or others. Any of the techniques for matching of words can similarly be used for matching of concepts. In further aspects, Similarity Comparison **125** can omit some of the words, characters, and/or other text from the comparison in determining substantial similarity of Object Properties **630**. In one example, rear-most words, characters, and/or other text can be omitted from comparison. In another example, shorter words and/or other text can be omitted from comparison. In general, any word, character, and/or other text can be omitted from comparison depending on implementation. Other techniques known in art for comparing words, characters, and/or other text can be utilized herein.

In some embodiments, Similarity Comparison **125** can compare one or more Extra Info **527** (i.e. time information, location information, computed information, visual information, acoustic information, contextual information, and/or other information, etc.) in addition to or instead of comparing Collections of Object Representations **525** or portions thereof in determining substantial similarity of Knowledge Cells **800**. Extra Info **527** can be set to be less, equally, or more important (i.e. as indicated by importance index [later described], etc.) than Collections of Object Representations **525**, Object Representations **625**, Object Properties **630**, and/or other elements in the comparison. Since Extra Info **527** may include any contextual or other information that can be useful in determining similarity of any compared elements, Extra Info **527** can be used to enhance any of the aforementioned similarity determinations as applicable.

In some embodiments, Similarity Comparison **125** can also compare one or more Instruction Sets **526** in addition to or instead of comparing Collections of Object Representations **525** or portions thereof in determining substantial similarity of Knowledge Cells **800**. In some aspects, Similarity Comparison **125** can compare portions of Instruction Sets **526** to determine substantial or other similarity of Instruction Sets **526**. Similar to the above-described thresholds, thresholds for the number or percentage of matching portions of the compared Instruction Sets **526** can be utilized in determining substantial or other similarity of the compared Instruction Sets **526**. Such thresholds can be defined by a user, by ACAAO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In other aspects, Similarity Comparison **125** can compare text (i.e. characters, words, phrases, etc.), numbers, or other data (i.e. bits, etc.) to determine substantial or other similarity of Instruction Sets **526**. Any other comparison technique can be utilized in comparing Instruction Sets **526** in alternate implementations. Instruction Sets **526** can be set to be less, equally, or more important (i.e. as indicated by importance index [later described], etc.) than Collections of Object Representations **525**, Object Representations **625**, Object Properties **630**, Extra Info **527**, and/or other elements in the comparison.

In some embodiments, an importance index (not shown) or other importance ranking technique can be used in any of the previously described comparisons or other processing involving elements of different importance. Importance index indicates importance of the element to or with which the index is assigned or associated. For example, importance index may indicate importance of a Knowledge Cell **800**, Collection of Object Representations **525**, Object Representation **625**, Object Property **630**, Instruction Set **526**, Extra Info **527**, and/or other element to or with which the index is assigned or associated. In some aspects, importance index on a scale from 0 to 1 can be utilized, although, any other range can also be utilized. Importance index can be stored in or associated with the element to which the index pertains. Importance indexes of various elements can be defined by a user, by ACAAO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. In one example, a higher Importance index can be assigned to more substantive or larger Collections of Object Representations **525** (i.e. Collections of Object Representations **525** comprising a higher number of Object Representations **625**, etc.). In another example, a higher importance index can be assigned to Object Representations **625** representing closer, larger, and/or other Objects **615**. Any importance index can be assigned to or associated with any element described herein depending on implementation. Any importance ranking technique can be utilized as or instead of importance index in alternate embodiments.

In some embodiments, Similarity Comparison **125** may generate a similarity index (not shown) for any of the compared elements. Similarity index indicates how well an element is matched with another element. For example, similarity index indicates how well a Knowledge Cell **800**, Collection of Object Representations **525**, Object Representation **625**, Object Property **630**, Instruction Set **526**, Extra Info **527**, and/or other element is matched with a compared element. In some aspects, similarity index on a scale from 0 to 1 can be utilized, although, any other range can also be utilized. Similarity index can be generated by Similarity Comparison **125** whether substantial or other similarity between the compared elements is achieved or not. In one example, similarity index can be determined for a Knowledge Cell **800** based on a ratio/percentage of matched or substantially matched Collections of Object Representations **525** relative to the number of Collections of Object Repre-

sentations **525** in the compared Knowledge Cell **800**. Specifically, similarity index of 0.89 is determined if 89% of Collections of Object Representations **525** of one Knowledge Cell **800** match or substantially match Collections of Object Representations **525** of another Knowledge Cell **800**. In some designs, importance (i.e. as indicated by importance index, etc.) of one or more Collections of Object Representations **525** can be included in the calculation of a weighted similarity index. Similar determination of similarity index can be implemented with Collections of Object Representations **525**, Object Representations **625**, Object Properties **630**, Instruction Sets **526**, Extra Info **527**, and/or other elements or portions thereof. Any combination of the aforementioned similarity index determinations or calculations can be utilized in alternate embodiments. Any similarity ranking technique can be utilized to determine or calculate similarity index in alternate embodiments.

Referring to FIG. **20**, an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Neural Network **530***a* comprising shortcut Connections **853** is illustrated. In some designs, Knowledge Cells **800** in one Layer **854** of Neural Network **530***a* can be connected with Knowledge Cells **800** in any Layer **854**, not only in a successive Layer **854**, thereby creating shortcuts (i.e. shortcut Connections **853**, etc.) through Neural Network **530***a*. In some aspects, creating a shortcut Connection **853** can be implemented by performing Similarity Comparisons **125** of a Knowledge Cell **800** from Knowledge Structuring Unit **520** with Knowledge Cells **800** in any Layer **854** when applying (i.e. storing, copying, etc.) the Knowledge Cell **800** from Knowledge Structuring Unit **520** onto Neural Network **530***a*. Once created, shortcut Connections **853** enable a wider variety of Knowledge Cells **800** to be considered when selecting a path through Neural Network **530***a*. In some embodiments, Knowledge Structuring Unit **520** structures or generates Knowledge Cells **800** and the system applies them onto Neural Network **530***a*, thereby implementing learning Avatar's **605** operation in circumstances including objects with various properties. The system can perform Similarity Comparisons **125** of a Knowledge Cell **800** from Knowledge Structuring Unit **520** with Knowledge Cells **800** in one or more Layers **854** of Neural Network **530***a*. If a substantially similar Knowledge Cell **800** is not found in the one or more Layers **854** of Neural Network **530***a*, the system may insert (i.e. copy, store, etc.) the Knowledge Cell **800** from Knowledge Structuring Unit **520** into a Layer **854** of Neural Network **530***a*, and create a Connection **853** to the inserted Knowledge Cell **800** from a prior Knowledge Cell **800** including assigning an occurrence count to the new Connection **853**, calculating a weight of the new Connection **853**, and updating any other Connections **853** originating from the prior Knowledge Cell **800**. On the other hand, if a substantially similar Knowledge Cell **800** is found in the one or more Layers **854** of Neural Network **530***a*, the system may update occurrence count and weight of Connection **853** to that Knowledge Cell **800** from a prior Knowledge Cell **800**, and update any other Connections **853** originating from the prior Knowledge Cell **800**. Any of the previously described and/or other techniques for comparing, inserting, updating, and/or other operations on Knowledge Cells **800**, Connections **853**, Layers **854**, and/or other elements can similarly be utilized in Neural Network **530***a* that comprises shortcut Connections **853**.

Referring to FIG. **21**, an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Graph **530***b* is illustrated. In some aspects, any Knowledge Cell **800** can be connected with any other Knowledge Cell **800** in Graph **530***b*. In other aspects, any Knowledge Cell **800** can be connected with itself and/or any other Knowledge Cell **800** in Graph **530***b*. In some embodiments, Knowledge Structuring Unit **520** structures or generates Knowledge Cells **800** and the system applies (i.e. store, copy, etc.) them onto Graph **530***b*, thereby implementing learning Avatar's **605** operation in circumstances including objects with various properties. The system can perform Similarity Comparisons **125** of a Knowledge Cell **800** from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Graph **530***b*. If a substantially similar Knowledge Cell **800** is not found in Graph **530***b*, the system may insert (i.e. copy, store, etc.) the Knowledge Cell **800** from Knowledge Structuring Unit **520** into Graph **530***b*, and create a Connection **853** to the inserted Knowledge Cell **800** from a prior Knowledge Cell **800** including assigning an occurrence count to the new Connection **853**, calculating a weight of the new Connection **853**, and updating any other Connections **853** originating from the prior Knowledge Cell **800**. On the other hand, if a substantially similar Knowledge Cell **800** is found in Graph **530***b*, the system may update occurrence count and weight of Connection **853** to that Knowledge Cell **800** from a prior Knowledge Cell **800**, and update any other Connections **853** originating from the prior Knowledge Cell **800**. Any of the previously described and/or other techniques for comparing, inserting, updating, and/or other operations on Knowledge Cells **800**, Connections **853**, and/or other elements can similarly be utilized in Graph **530***b*.

For example, the system can perform Similarity Comparisons **125** of Knowledge Cell **800***ba* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Graph **530***b*. In the case that a substantially similar match is not found, the system may insert Knowledge Cell **800***ha* into Graph **530***b* and copy Knowledge Cell **800***ba* into the inserted Knowledge Cell **800***ha*. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bb* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Graph **530***b*. In the case that a substantially similar match is found between Knowledge Cell **800***bb* and Knowledge Cell **800***hb*, the system may create Connection **853***h***1** between Knowledge Cell **800***ha* and Knowledge Cell **800***hb* with occurrence count of 1 and weight of 1. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bc* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Graph **530***b*. In the case that a substantially similar match is found between Knowledge Cell **800***bc* and Knowledge Cell **800***hc*, the system may update occurrence count and weight of Connection **853***h***2** between Knowledge Cell **800***hb* and Knowledge Cell **800***hc*, and update weights of other outgoing Connections **853** (one in this example) originating from Knowledge Cell **800***hb* as previously described. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bd* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Graph **530***b*. In the case that a substantially similar match is not found, the system may insert Knowledge Cell **800***hd* into Graph **530***b* and copy Knowledge Cell **800***bd* into the inserted Knowledge Cell **800***hd*. The system may also create Connection **853***h***3** between Knowledge Cell **800***hc* and Knowledge Cell **800***hd* with occurrence count of 1 and weight calculated based on the occurrence count as previously described. The system may also update weights of other outgoing Connections **853** (one in this example)

originating from Knowledge Cell **800***hc* as previously described. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***be* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Graph **530***b*. In the case that a substantially similar match is not found, the system may insert Knowledge Cell **800***he* into Graph **530***b* and copy Knowledge Cell **800***be* into the inserted Knowledge Cell **800***he*. The system may also create Connection **853***h***4** between Knowledge Cell **800***hd* and Knowledge Cell **800***he* with occurrence count of 1 and weight of 1. Applying any additional Knowledge Cells **800** from Knowledge Structuring Unit **520** onto Graph **530***b* follows similar logic or process as the above-described.

Referring to FIG. **22**, an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Collection of Sequences **530***c* is illustrated. Collection of Sequences **530***c* comprises the functionality for storing one or more Sequences **533**. Sequence **533** comprises the functionality for storing any number of Knowledge Cells **800**. For example, Knowledge Structuring Unit **520** structures or generates Knowledge Cells **800** and the system applies them onto Collection of Sequences **530***c*, thereby implementing learning Avatar's **605** operation in circumstances including objects with various properties. The system can perform collective Similarity Comparisons **125** of Knowledge Cells **800** from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Sequences **533** of Collection of Sequences **530***c* to find a Sequence **533** comprising Knowledge Cells **800** that are collectively substantially similar to the Knowledge Cells **800** from Knowledge Structuring Unit **520**. If Sequence **533** comprising such collectively substantially similar Knowledge Cells **800** is not found in Collection of Sequences **530***c*, the system may create a new Sequence **533** comprising the Knowledge Cells **800** from Knowledge Structuring Unit **520** and insert (i.e. copy, store, etc.) the new Sequence **533** into Collection of Sequences **530***c*. On the other hand, if Sequence **533** comprising collectively substantially similar Knowledge Cells **800** is found in Collection of Sequences **530***c*, the system may optionally omit inserting the Knowledge Cells **800** from Knowledge Structuring Unit **520** into Collection of Sequences **530***c* as inserting a similar Sequence **533** may not add much or any additional knowledge. This approach can save storage resources and limit the number of Knowledge Cells **800** that may later need to be processed or compared. In some aspects, a Sequence **533** may include Knowledge Cells **800** relating to a single operation of Avatar **605**. In other aspects, a Sequence **533** may include Knowledge Cells **800** relating to a part of an operation of Avatar **605**. In further aspects, one or more long Sequences **533** each including Knowledge Cells **800** of multiple operations of Avatar **605** can be utilized. In one example, Knowledge Cells **800** of all operations can be stored in a single long Sequence **533** in which case Collection of Sequences **530***c* as a separate element can be omitted. In another example, Knowledge Cells **800** of multiple operations can be included in a plurality of long Sequences **533** such as hourly, daily, weekly, monthly, yearly, or other periodic or other Sequences **533**. Similarity Comparisons **125** can be performed by traversing the one or more long Sequences **533** to find a match or substantially similar match. For instance, the system can perform collective Similarity Comparisons **125** of Knowledge Cells **800** from Knowledge Structuring Unit **520** with Knowledge Cells **800** in subsequences of a long Sequence **533** in incremental or other traversing pattern to find a subsequence comprising Knowledge Cells **800** that are collectively substantially similar to the Knowledge Cells **800** from Knowledge Structuring Unit **520**. The incremental traversing pattern may start from one end of a long Sequence **533** and move the comparison subsequence up or down one or any number of incremental Knowledge Cells **800** at a time. Other traversing patterns or methods can be employed such as starting from the middle of the Sequence **533** and subdividing the resulting sub-sequences in a recursive pattern, or any other traversing pattern or method. If a subsequence comprising collectively substantially similar Knowledge Cells **800** is not found in the long Sequence **533**, the system may concatenate or append the Knowledge Cells **800** from Knowledge Structuring Unit **520** to the long Sequence **533**. In further aspects, Connections **853** can optionally be used in Sequence **533** to connect Knowledge Cells **800**. For example, a Knowledge Cell **800** can be connected not only with a next Knowledge Cell **800** in the Sequence **533**, but also with any other Knowledge Cell **800** in the Sequence **533**, thereby creating alternate routes or shortcuts through the Sequence **533**. Any number of Connections **853** connecting any Knowledge Cells **800** can be utilized. Any of the previously described and/or other techniques for comparing, inserting, updating, and/or other operations on Knowledge Cells **800**, Connections **853**, and/or other elements can similarly be utilized in Sequences **533** and/or Collection of Sequences **530***c*.

In some embodiments, various elements and/or techniques can be utilized in the aforementioned substantial similarity determinations with respect to collectively compared Knowledge Cells **800** and/or other elements. In some aspects, substantial similarity of collectively compared Knowledge Cells **800** can be determined based on similarities or similarity indexes of the individually compared Knowledge Cells **800**. In one example, an average of similarities or similarity indexes of individually compared Knowledge Cells **800** can be used to determine similarity of collectively compared Knowledge Cells **800**. In another example, a weighted average of similarities or similarity indexes of individually compared Knowledge Cells **800** can be used to determine similarity of collectively compared Knowledge Cells **800**. For instance, to affect the weighting of collective similarity, a higher weight or importance (i.e. importance index, etc.) can be assigned to the similarities or similarity indexes of some Knowledge Cells **800** and lower for other Knowledge Cells **800**. Any higher or lower weight or importance assignment can be implemented. In other aspects, any of the previously described or other thresholds for substantial similarity of individually compared elements can similarly be utilized for collectively compared elements. In one example, substantial similarity of collectively compared Knowledge Cells **800** can be achieved when their collective similarity or similarity index exceeds a similarity threshold. In another example, substantial similarity of collectively compared Knowledge Cells **800** can be achieved when at least a threshold number or percentage of Knowledge Cells **800** from the collectively compared Knowledge Cells **800** match or substantially match. Similarly, substantial similarity of collectively compared Knowledge Cells **800** can be achieved when a number or percentage of matching or substantially matching Knowledge Cells **800** from the collectively compared Knowledge Cells **800** exceeds a threshold. Such thresholds can be defined by a user, by ACAAO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. Similar elements and/or techniques as the aforementioned can be

used for similarity determinations of other collectively compared elements such as Collections of Object Representations **525**, Object Representations **625**, Object Properties **630**, Instruction Sets **526**, Extra Info **527**, and/or others. Similarity determinations of collectively compared elements may include any features, functionalities, and embodiments of Similarity Comparison **125**, and vice versa.

Any of the previously described data structures or arrangements of Knowledge Cells **800** such as Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, and/or others can be used alone, or in combination with each other or with other elements, in alternate embodiments. In one example, a path in Neural Network **530***a* or Graph **530***b* may include its own separate sequence of Knowledge Cells **800** that are not interconnected with Knowledge Cells **800** in other paths. In another example, a part of a path in Neural Network **530***a* or Graph **530***b* may include a sequence of Knowledge Cells **800** interconnected with Knowledge Cells **800** in other paths, whereas, another part of the path may include its own separate sequence of Knowledge Cells **800** that are not interconnected with Knowledge Cells **800** in other paths. Any other combinations or arrangements of Knowledge Cells **800** can be implemented.

Referring to FIG. **23**, an embodiment of determining anticipatory Instruction Sets **526** from a single Knowledge Cell **800** is illustrated. Knowledge Cell **800** may be part of a Knowledgebase **530** (i.e. Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.) such as Collection of Knowledge Cells **530***d*. Decision-making Unit **540** comprises the functionality for anticipating or determining Avatar's **605** operation in circumstances including objects with various properties. Decision-making Unit **540** comprises the functionality for anticipating or determining Instruction Sets **526** to be used or executed in Avatar's **605** autonomous operation. In some aspects, Instruction Sets **526** anticipated or determined to be used or executed in Avatar's **605** autonomous operation may be referred to as anticipatory Instruction Sets **526**, alternate Instruction Sets **526**, and/or other suitable name or reference. Therefore, these terms can be used interchangeably herein depending on context. Decision-making Unit **540** also comprises other disclosed functionalities.

In some aspects, Decision-making Unit **540** may anticipate or determine Instruction Sets **526** (i.e. anticipatory Instruction Sets **526**, etc.) for autonomous Avatar **605** operation by performing Similarity Comparisons **125** of incoming Collections of Object Representations **525** or portions thereof from Object Processing Unit **140** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Knowledgebase **530** (i.e. Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.). A Knowledge Cell **800** includes knowledge (i.e. one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**, etc.) of how Avatar **605** operated in a circumstance including objects with various properties as previously described. When one or more Collections of Object Representations **525** representing objects with similar properties are received in the future, Decision-making Unit **540** can anticipate the Instruction Sets **526** (i.e. anticipatory Instruction Sets **526**, etc.) previously learned in a similar circumstance, thereby enabling autonomous Avatar **605** operation. In some aspects, Decision-making Unit **540** can perform Similarity Comparisons **125** of incoming Collections of Object Representations **525** from Object Processing Unit **140** with Collections of Object Representations **525** from Knowledge Cells **800** in Knowledgebase **530** (i.e. Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.). If one or more substantially similar Collections of Object Representations **525** or portions thereof are found in a Knowledge Cell **800** from Knowledgebase **530**, Instruction Sets **526** (i.e. anticipatory Instruction Sets **526**, etc.) for autonomous Avatar **605** operation can be anticipated in Instruction Sets **526** correlated with the one or more Collections of Object Representations **525** from the Knowledge Cell **800**. In some designs, subsequent one or more Instruction Sets **526** for autonomous Avatar **605** operation can be anticipated in Instruction Sets **526** correlated with subsequent Collections of Object Representations **525** from the Knowledge Cell **800** or other Knowledge Cells **800**, thereby anticipating not only current, but also additional future Instruction Sets **526**. Although, Extra Info **527** is not shown in this and/or other figures for clarity of illustration, it should be noted that any Collection of Object Representations **525**, Instruction Set **526**, and/or other element may include or be associated with Extra Info **527** and that Decision-making Unit **540** can utilize Extra Info **527** for enhanced decision making.

For example, Decision-making Unit **540** can perform Similarity Comparison **125** of Collection of Object Representations **525/1** or portions thereof from Object Processing Unit **140** with Collection of Object Representations **525***a***1** or portions thereof from Knowledge Cell **800***oa*. Collection of Object Representations **525***a***1** or portions thereof from Knowledge Cell **800***oa* may be found substantially similar. Decision-making Unit **540** can anticipate Instruction Sets **526***a***1**-**526***a***3** correlated with Collection of Object Representations **525***a***1**, thereby enabling autonomous Avatar **605** operation. Decision-making Unit **540** can then perform Similarity Comparison **125** of Collection of Object Representations **52512** or portions thereof from Object Processing Unit **140** with Collection of Object Representations **525***a***2** or portions thereof from Knowledge Cell **800***oa*. Collection of Object Representations **525***a***2** or portions thereof from Knowledge Cell **800***oa* may be found substantially similar. Decision-making Unit **540** can anticipate Instruction Set **526***a***4** correlated with Collection of Object Representations **525***a***2**, thereby enabling autonomous Avatar **605** operation. Decision-making Unit **540** can then perform Similarity Comparison **125** of Collection of Object Representations **52513** or portions thereof from Object Processing Unit **140** with Collection of Object Representations **525***a***3** or portions thereof from Knowledge Cell **800***oa*. Collection of Object Representations **525***a***3** or portions thereof from Knowledge Cell **800***oa* may be found substantially similar. Decision-making Unit **540** may not anticipate any Instruction Sets **526** since none are correlated with Collection of Object Representations **525***a***3**. Decision-making Unit **540** can then perform Similarity Comparison **125** of Collection of Object Representations **52514** or portions thereof from Object Processing Unit **140** with Collection of Object Representations **525***a***4** or portions thereof from Knowledge Cell **800***oa*. Collection of Object Representations **525***a***4** or portions thereof from Knowledge Cell **800***oa* may not be found substantially similar. Decision-making Unit **540** can then perform Similarity Comparison **125** of Collection of Object Representations **52515** or portions thereof from Object Processing Unit **140** with Collection of Object Representations **525***a***5** or portions thereof from Knowledge Cell **800***oa*. Collection of Object Representations **525***a***5** or portions thereof from Knowledge Cell **800***oa* may not be found

substantially similar. Decision-making Unit **540** can implement similar logic or process for any additional Collections of Object Representations **525** from Object Processing Unit **140**, and so on.

It should be understood that any of the described elements and/or techniques in the foregoing example can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells **800** or elements (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof would be affected accordingly. In one example, Extra Info **527** can be included in the Similarity Comparisons **125** as previously described. In another example, as history of incoming Collections of Object Representations **525** becomes available, Decision-making Unit **540** can perform collective Similarity Comparisons **125** of the history of Collections of Object Representations **525** or portions thereof from Object Processing Unit **140** with subsequences of Collections of Object Representations **525** or portions thereof from Knowledge Cell **800**. In a further example, the described comparisons in a single Knowledge Cell **800** may be performed on any number of Knowledge Cells **800** sequentially or in parallel. Parallel processors such as a plurality of Processors **11** or cores thereof can be utilized for such parallel processing. In a further example, various arrangements of Collections of Object Representations **525** and/or other elements in a Knowledge Cell **800** can be utilized as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to FIG. **24**, an embodiment of determining anticipatory Instruction Sets **526** by traversing a single Knowledge Cell **800** is illustrated. Knowledge Cell **800** may be part of a Knowledgebase **530** (i.e. Neural Network **530*a***, Graph **530*b***, Collection of Sequences **530*c***, Sequence **533**, Collection of Knowledge Cells **530*d***, etc.) such as Collection of Knowledge Cells **530*d***. For example, Decision-making Unit **540** can perform Similarity Comparison **125** of Collection of Object Representations **525/1** or portions thereof from Object Processing Unit **140** with Collection of Object Representations **525*a*1** or portions thereof from Knowledge Cell **800*oa***. Collection of Object Representations **525*a*1** or portions thereof from Knowledge Cell **800*oa*** may not be found substantially similar. Decision-making Unit **540** can then perform Similarity Comparison **125** of Collection of Object Representations **525/1** or portions thereof from Object Processing Unit **140** with Collection of Object Representations **525*a*2** or portions thereof from Knowledge Cell **800*oa***. Collection of Object Representations **525*a*2** or portions thereof from Knowledge Cell **800*oa*** may not be found substantially similar. Decision-making Unit **540** can then perform Similarity Comparison **125** of Collection of Object Representations **525/1** or portions thereof from Object Processing Unit **140** with Collection of Object Representations **525*a*3** or portions thereof from Knowledge Cell **800*oa***. Collection of Object Representations **525*a*3** or portions thereof from Knowledge Cell **800*oa*** may be found substantially similar. Decision-making Unit **540** may not anticipate any Instruction Sets **526** since none are correlated with Collection of Object Representations **525*a*3**. Decision-making Unit **540** can then perform Similarity Comparison **125** of Collection of Object Representations **52512** or portions thereof from Object Processing Unit **140** with Collection of Object Representations **525*a*4** or portions thereof from Knowledge Cell **800*oa***. Collection of Object Representations **525*a*4** or portions thereof from Knowledge Cell **800*oa*** may be found substantially similar. Decision-making Unit **540** can anticipate Instruction Sets **526*a*5-526*a*6** correlated with Collection of Object Representations **525*a*4**, thereby enabling autonomous Avatar **605** operation. Decision-making Unit **540** can then perform Similarity Comparison **125** of Collection of Object Representations **52513** or portions thereof from Object Processing Unit **140** with Collection of Object Representations **525*a*5** or portions thereof from Knowledge Cell **800*oa***. Collection of Object Representations **525*a*5** or portions thereof from Knowledge Cell **800*oa*** may be found substantially similar. Decision-making Unit **540** may not anticipate any Instruction Sets **526** since none are correlated with Collection of Object Representations **525*a*5**. Decision-making Unit **540** can implement similar logic or process for any additional Collections of Object Representations **525** from Object Processing Unit **140**, and so on.

It should be understood that any of the described elements and/or techniques in the foregoing example can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells **800** or elements (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof would be affected accordingly. In one example, Extra Info **527** can be included in the Similarity Comparisons **125** as previously described. In another example, as history of incoming Collections of Object Representations **525** becomes available, Decision-making Unit **540** can perform collective Similarity Comparisons **125** of the history of Collections of Object Representations **525** or portions thereof from Object Processing Unit **140** with subsequences of Collections of Object Representations **525** or portions thereof from Knowledge Cell **800**. In a further example, traversing may be performed in incremental traversing pattern such as starting from one end of Knowledge Cell **800** and moving the comparison subsequence up or down the list one or any number of incremental Collections of Object Representations **525** at a time. Other traversing patterns or methods can be employed such as starting from the middle of the Knowledge Cell **800** and subdividing the resulting subsequence in a recursive pattern, or any other traversing pattern or method. In a further example, the described traversing of a single Knowledge Cell **800** may be performed on any number of Knowledge Cells **800** sequentially or in parallel. Parallel processors such as a plurality of Processors **11** or cores thereof can be utilized for such parallel processing. In a further example, various arrangements of Collections of Object Representations **525** and/or other elements in a Knowledge Cell **800** can be utilized as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to FIG. **25**, an embodiment of determining anticipatory Instruction Sets **526** using collective similarity comparisons is illustrated. For example, Decision-making Unit **540** can perform Similarity Comparisons **125** of Collection of Object Representations **525/1** or portions thereof from Object Processing Unit **140** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Collection of Knowledge Cells **530*d***. Collection of Object Representations **525*c*1** or portions thereof from Knowledge Cell **800*rc*** may be found substantially

similar with highest similarity. Decision-making Unit **540** can anticipate any Instruction Sets **526** (not shown) correlated with Collection of Object Representations **525**c**1**, thereby enabling autonomous Avatar **605** operation. Decision-making Unit **540** can then perform collective Similarity Comparisons **125** of Collections of Object Representations **525/1-52512** or portions thereof from Object Processing Unit **140** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Collection of Knowledge Cells **530**d. Collections of Object Representations **525**c**1-525**c**2** or portions thereof from Knowledge Cell **800**rc may be found substantially similar with highest similarity. Decision-making Unit **540** can anticipate any Instruction Sets **526** (not shown) correlated with Collection of Object Representations **525**c**2**, thereby enabling autonomous Avatar **605** operation. Decision-making Unit **540** can then perform collective Similarity Comparisons **125** of Collections of Object Representations **525/1-52513** or portions thereof from Object Processing Unit **140** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Collection of Knowledge Cells **530**d. Collections of Object Representations **525**d**1-525**d**3** or portions thereof from Knowledge Cell **800**rd may be found substantially similar with highest similarity. Decision-making Unit **540** can anticipate any Instruction Sets **526** (not shown) correlated with Collection of Object Representations **525**d**3**, thereby enabling autonomous Avatar **605** operation. Decision-making Unit **540** can then perform collective Similarity Comparisons **125** of Collections of Object Representations **525/1-52514** or portions thereof from Object Processing Unit **140** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Collection of Knowledge Cells **530**d. Collections of Object Representations **525**d**1-525**d**4** or portions thereof from Knowledge Cell **800**rd may be found substantially similar with highest similarity. Decision-making Unit **540** can anticipate any Instruction Sets **526** (not shown) correlated with Collection of Object Representations **525**d**4**, thereby enabling autonomous Avatar **605** operation. Decision-making Unit **540** can then perform collective Similarity Comparisons **125** of Collections of Object Representations **525/1-52515** or portions thereof from Object Processing Unit **140** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Collection of Knowledge Cells **530**d. Collections of Object Representations **525**d**1-525**d**5** or portions thereof from Knowledge Cell **800**rd may be found substantially similar with highest similarity. Decision-making Unit **540** can anticipate any Instruction Sets **526** (not shown) correlated with Collection of Object Representations **525**d**5**, thereby enabling autonomous Avatar **605** operation. Decision-making Unit **540** can implement similar logic or process for any additional Collections of Object Representations **525** from Object Processing Unit **140**, and so on.

In some embodiments, various elements and/or techniques can be utilized in the aforementioned similarity determinations with respect to collectively compared Collections of Object Representations **525** and/or other elements. In some aspects, similarity of collectively compared Collections of Object Representations **525** can be determined based on similarities or similarity indexes of the individually compared Collections of Object Representations **525**. In one example, an average of similarities or similarity indexes of individually compared Collections of Object Representations **525** can be used to determine similarity of collectively compared Collections of Object Representations **525**. In another example, a weighted average of similarities or similarity indexes of individually compared Collections of Object Representations **525** can be used to determine similarity of collectively compared Collections of Object Representations **525**. For instance, to affect the weighting of collective similarity, a higher weight or importance (i.e. importance index, etc.) can be assigned to the similarities or similarity indexes of some (i.e. more substantive or larger, etc.) Collections of Object Representations **525** and lower for other (i.e. less substantive or smaller, etc.) Collections of Object Representations **525**. Any other higher or lower weight or importance assignment can be implemented. In other aspects, any of the previously described or other thresholds for substantial similarity of individually compared elements can be similarly utilized for collectively compared elements. In one example, substantial similarity of collectively compared Collections of Object Representations **525** can be achieved when their collective similarity or similarity index exceeds a similarity threshold. In another example, substantial similarity of collectively compared Collections of Object Representations **525** can be achieved when at least a threshold number or percentage of Collections of Object Representations **525** or portions thereof from the collectively compared Collections of Object Representations **525** match or substantially match. Similarly, substantial similarity of collectively compared Collections of Object Representations **525** can be achieved when a number or percentage of matching or substantially matching Collections of Object Representations **525** or portions thereof from the collectively compared Collections of Object Representations **525** exceeds a threshold. Such thresholds can be defined by a user, by ACAAO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. Similar elements and/or techniques as the aforementioned can be used for similarity determinations of other collectively compared elements such as Object Representations **625**, Object Properties **630**, Instruction Sets **526**, Extra Info **527**, Knowledge Cells **800**, and/or others. Similarity determinations of collectively compared elements may include any features, functionalities, and embodiments of Similarity Comparison **125**, and vice versa.

It should be understood that any of the described elements and/or techniques in the foregoing example can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells **800** or elements (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info **527** in Similarity Comparisons **125**, traversing of Knowledge Cells **800** or other elements, using history of Collections of Object Representations **525** or Knowledge Cells **800** for collective Similarity Comparisons **125**, using various arrangements of Collections of Object Representations **525** and/or other elements in a Knowledge Cell **800**, and/or others can similarly be utilized in this example. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to FIG. **26**, an embodiment of determining anticipatory Instruction Sets **526** using Neural Network **530**a is illustrated. In some aspects, determining anticipatory Instruction Sets **526** using Neural Network **530**a may include selecting a path of Knowledge Cells **800** or elements

(i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof through Neural Network **530***a*. Decision-making Unit **540** can utilize various elements and/or techniques for selecting a path through Neural Network **530***a*. Although, these elements and/or techniques are described with respect to Neural Network **530***a* below, they can similarly be used in any Knowledgebase **530** (i.e. Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.) as applicable.

In some embodiments, Decision-making Unit **540** can utilize similarity index in selecting Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof in a path through Neural Network **530***a*. For instance, similarity index may indicate how well one Knowledge Cell **800** or portions thereof are matched with another Knowledge Cell **800** or portions thereof as previously described. In one example, Decision-making Unit **540** may select a Knowledge Cell **800** comprising one or more Collections of Object Representations **525** with highest similarity index even if Connection **853** pointing to that Knowledge Cell **800** has less than the highest weight. Therefore, similarity index or other such element or parameter can override or disregard the weight of a Connection **853** or other element. In another example, Decision-making Unit **540** may select a Knowledge Cell **800** comprising one or more Collections of Object Representations **525** whose similarity index is higher than or equal to a weight of Connection **853** pointing to that Knowledge Cell **800**. In a further example, Decision-making Unit **540** may select a Knowledge Cell **800** comprising one or more Collections of Object Representations **525** whose similarity index is lower than or equal to a weight of Connection **853** pointing to that Knowledge Cell **800**. Similarity index can be set to be more, less, or equally important than a weight of a Connection **853**.

In some embodiments, Decision-making Unit **540** can utilize Connections **853** in selecting Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof in a path through Neural Network **530***a*. In some aspects, Decision-making Unit **540** can take into account weights of Connections **853** among the interconnected Knowledge Cells **800** in choosing from which Knowledge Cell **800** to compare one or more Collections of Object Representations **525** first, second, third, and so on. Specifically, for instance, Decision-making Unit **540** can perform Similarity Comparisons **125** with one or more Collections of Object Representations **525** from Knowledge Cell **800** pointed to by the highest weight Connection **853** first, Collections of Object Representations **525** from Knowledge Cell **800** pointed to by the second highest weight Connection **853** second, and so on. In other aspects, Decision-making Unit **540** can stop performing Similarity Comparisons **125** as soon as it finds one or more substantially similar Collections of Object Representations **525** in an interconnected Knowledge Cell **800**. In further aspects, Decision-making Unit **540** may only follow the highest weight Connection **853** to arrive at a Knowledge Cell **800** comprising one or more Collections of Object Representations **525** to be compared, thereby disregarding Connections **853** with less than the highest weight. In further aspects, Decision-making Unit **540** may ignore weights and/or other parameters of Connections **853**. In further aspects, Decision-making Unit **540** may ignore Connections **853**.

In some embodiments, Decision-making Unit **540** can utilize a bias to adjust similarity index, weight of a Connection **853**, and/or other element or parameter used in selecting Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof in a path through Neural Network **530***a*. In one example, Decision-making Unit **540** may select a Knowledge Cell **800** comprising one or more Collections of Object Representations **525** whose similarity index multiplied by or adjusted for a bias is higher than or equal to a weight of Connection **853** pointing to that Knowledge Cell **800**. In another example, Decision-making Unit **540** may select a Knowledge Cell **800** comprising one or more Collections of Object Representations **525** whose similarity index multiplied by or adjusted for a bias is lower than or equal to a weight of Connection **853** pointing to that Knowledge Cell **800**. In a further example, bias can be used to resolve deadlock situations where similarity index is equal to a weight of a Connection **853**. In some aspects, bias can be expressed in percentages such as 0.3 percent, 1.2 percent, 25.7 percent, 79.8 percent, 99.9 percent, 100.1 percent, 155.4 percent, 298.6 percent, 1105.5 percent, and so on. For example, a bias below 100 percent decreases an element or parameter to which it is applied, a bias equal to 100 percent does not change the element or parameter to which it is applied, and a bias higher than 100 percent increases the element or parameter to which it is applied. In general, any amount of bias can be utilized depending on implementation. Bias can be applied to one or more of a weight of a Connection **853**, similarity index, any other element or parameter, and/or all or any combination of them. Also, different biases can be applied to each of a weight of a Connection **853**, similarity index, or any other element or parameter. For example, 30 percent bias can be applied to similarity index and 15 percent bias can be applied to a weight of a Connection **853**. Also, different biases can be applied to various Layers **854** of Neural Network **530***a*, and/or other disclosed elements. Bias can be defined by a user, by ACAAO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input.

Any other element and/or technique can be utilized in selecting Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof in a path through Neural Network **530***a*.

In some embodiments, Neural Network **530***a* may include knowledge (i.e. interconnected Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**, etc.) of how Avatar **605** operated in circumstances including objects with various properties. In some aspects, determining anticipatory Instruction Sets **526** using Neural Network **530***a* may include selecting a path of Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof through Neural Network **530***a*. Individual and/or collective Similarity Comparisons **125** can be used to determine substantial similarity of the individually and/or collectively compared Collections of Object Representations **525** or portions thereof. Substantial similarity may be used primarily for selecting a path through Neural Network **530***a*, whereas, weight of any Connection **853** may be used secondarily or not at all.

For example, Decision-making Unit **540** can perform Similarity Comparisons **125** of Collections of Object Representations **525***a***1**-**525***an* or portions thereof from Object Processing Unit **140** with Collections of Object Representations **525** or portions thereof from one or more Knowledge Cells **800** in Layer **854***a* (or any other one or more Layers **854**, etc.). Collections of Object Representations **525** or

portions thereof from Knowledge Cell **800***ta* may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Avatar **605** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525***b***1**-**525***bn* or portions thereof from Object Processing Unit **140** with Collections of Object Representations **525** or portions thereof from one or more Knowledge Cells **800** in Layer **854***b* interconnected with Knowledge Cell **800***ta*. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***th* may be found collectively substantially similar with highest similarity, thus, Decision-making Unit **540** may follow Connection **853***t***1** disregarding its less than highest weight. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Avatar **605** operation. Since Connection **853***t***2** is the only connection from Knowledge Cell **800***thb*, Decision-making Unit **540** may follow Connection **853***t***2** and perform Similarity Comparisons **125** of Collections of Object Representations **525***c***1**-**525***cn* or portions thereof from Object Processing Unit **140** with Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***tc* in Layer **854***c*. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***tc* may be found collectively substantially similar. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Avatar **605** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525***d***1**-**525***dn* or portions thereof from Object Processing Unit **140** with Collections of Object Representations **525** or portions thereof from one or more Knowledge Cells **800** in Layer **854***d* interconnected with Knowledge Cell **800***tc*. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***td* may be found collectively substantially similar with highest similarity, thus, Decision-making Unit **540** may follow Connection **853***t***3**. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Avatar **605** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525***e***1**-**525***en* or portions thereof from Object Processing Unit **140** with Collections of Object Representations **525** or portions thereof from one or more Knowledge Cells **800** in Layer **854***e* interconnected with Knowledge Cell **800***td*. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***te* may be found collectively substantially similar with highest similarity, thus, Decision-making Unit **540** may follow Connection **853***t***4**. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Avatar **605** operation. Decision-making Unit **540** can implement similar logic or process for any additional Collections of Object Representations **525** from Object Processing Unit **140**.

The foregoing exemplary embodiment provides an example of utilizing a combination of collective Similarity Comparisons **125**, individual Similarity Comparisons **125**, Connections **853**, and/or other elements or techniques. It should be understood that any of these elements and/or techniques can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof in a path through Neural Network **530***a* would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info **527** in Similarity Comparisons **125**, traversing of Knowledge Cells **800** or other elements, using history of Collections of Object Representations **525** or Knowledge Cells **800** for collective Similarity Comparisons **125**, using various arrangements of Collections of Object Representations **525** and/or other elements in a Knowledge Cell **800**, and/or others can similarly be utilized in this example. These elements and/or techniques can similarly be utilized in Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, and/or other data structures or arrangements. In some aspects, instead of anticipating Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525**, Decision-making Unit **540** can anticipate instruction Sets **526** correlated with substantially similar streams of Collections of Object Representations **525**. In other aspects, any time that substantial similarity or other similarity threshold is not achieved in compared Collections of Object Representations **525** or portions thereof from any of the Knowledge Cells **800**, Decision-making Unit **540** can decide to look for a substantially or otherwise similar Collections of Object Representations **525** or portions thereof in Knowledge Cells **800** elsewhere in Neural Network **530***a* such as in any Layer **854** subsequent to a current Layer **854**, in the first Layer **854**, in the entire Neural Network **530***a*, and/or others, even if such Knowledge Cell **800** may be unconnected with a prior Knowledge Cell **800**. It should be noted that any of Collections of Object Representations **525***a***1**-**525***an*, **525***b***1**-**525***bn*, **525***c***1**-**525***cn*, **525***d***1**-**525***dn*, **525***e***1**-**525***en*, etc. may include one Collection of Object Representations **525** or a stream of Collections of Object Representations **525**. It should also be noted that any Knowledge Cell **800** may include one Collection of Object Representations **525** or a stream of Collections of Object Representations **525** as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to FIG. **27**, an embodiment of determining anticipatory Instruction Sets **526** using Graph **530***b* is illustrated. Graph **530***b* may include knowledge (i.e. interconnected Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**, etc.) of how

Avatar **605** operated in circumstances including objects with various properties. In some aspects, determining anticipatory Instruction Sets **526** using Graph **530***b* may include selecting a path of Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof through Graph **530***b*. Individual and/or collective Similarity Comparisons **125** can be used to determine substantial similarity of the individually and/or collectively compared Collections of Object Representations **525** or portions thereof. Substantial similarity may be used primarily for selecting a path through Graph **530***b*, whereas, weight of any Connection **853** may be used secondarily or not at all.

For example, Decision-making Unit **540** can perform Similarity Comparisons **125** of Collections of Object Representations **525***a***1**-**525***an* or portions thereof from Object Processing Unit **140** with Collections of Object Representations **525** or portions thereof from one or more Knowledge Cells **800** in Graph **530***b*. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***ua* may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Avatar **605** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525***b***1**-**525***bn* or portions thereof from Object Processing Unit **140** with Collections of Object Representations **525** or portions thereof from one or more Knowledge Cells **800** in Graph **530***b* interconnected with Knowledge Cell **800***ua* by outgoing Connections **853**. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***ub* may be found collectively substantially similar with highest similarity, thus, Decision-making Unit **540** may follow Connection **853***u***1** disregarding its less than highest weight. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Avatar **605** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525***c***1**-**525***cn* or portions thereof from Object Processing Unit **140** with Collections of Object Representations **525** or portions thereof from one or more Knowledge Cells **800** in Graph **530***b* interconnected with Knowledge Cell **800***ub* by outgoing Connections **853**. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***uc* may be found collectively substantially similar with highest similarity, thus, Decision-making Unit **540** may follow Connection **853***u***2** disregarding its less than highest weight. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Avatar **605** operation. Since Connection **853***u***3** is the only connection from Knowledge Cell **800***uc*, Decision-making Unit **540** may follow Connection **853***u***3** and perform Similarity Comparisons **125** of Collections of Object Representations **525***d***1**-**525***dn* or portions thereof from Object Processing Unit **140** with Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***ud* in Graph **530***b*. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***ud* may be found collectively substantially similar. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Avatar **605** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525***e***1**-**525***en* or portions thereof from Object Processing Unit **140** with Collections of Object Representations **525** or portions thereof from one or more Knowledge Cells **800** in Graph **530***b* interconnected with Knowledge Cell **800***ud* by outgoing Connections **853**. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***ue* may be found collectively substantially similar with highest similarity, thus, Decision-making Unit **540** may follow Connection **853***u***4**. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Avatar **605** operation. Decision-making Unit **540** can implement similar logic or process for any additional Collections of Object Representations **525** from Object Processing Unit **140**.

The foregoing exemplary embodiment provides an example of utilizing a combination of collective Similarity Comparisons **125**, individual Similarity Comparisons **125**, Connections **853**, and/or other elements or techniques. It should be understood that any of these elements and/or techniques can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof in a path through Graph **530***b* would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info **527** in Similarity Comparisons **125**, traversing of Knowledge Cells **800** or other elements, using history of Collections of Object Representations **525** or Knowledge Cells **800** in collective Similarity Comparisons **125**, using various arrangements of Collections of Object Representations **525** and/or other elements in a Knowledge Cell **800**, and/or others can similarly be utilized in this example. These elements and/or techniques can similarly be utilized in Neural Network **530***a*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, and/or other data structures or arrangements. In some aspects, instead of anticipating Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525**, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially matching streams of Collections of Object Representations **525**. In other aspects, any time that substantial similarity or other similarity threshold is not achieved in compared Collections of Object Representations **525** or portions thereof of any of the Knowledge Cells **800**, Decision-making Unit **540** can decide to look for a substantially or otherwise similar Collections of Object Representations **525** or portions thereof in Knowledge Cells **800** elsewhere in Graph **530***b* even if such Knowledge Cell **800** may be unconnected with a prior Knowledge Cell **800**.

It should be noted that any of Collections of Object Representations **525*a*1**-**525*an***, **525*b*1**-**525*bn***, **525*c*1**-**525*cn***, **525*d*1**-**525*dn***, **525*e*1**-**525*en***, etc. may include one Collection of Object Representations **525** or a stream of Collections of Object Representations **525**. It should also be noted that any Knowledge Cell **800** may include one Collection of Object Representations **525** or a stream of Collections of Object Representations **525** as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to FIG. **28**, an embodiment of determining anticipatory Instruction Sets **526** using Collection of Sequences **530*c*** is illustrated. Collection of Sequences **530*c*** may include knowledge (i.e. sequences of Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**, etc.) of how Avatar **605** operated in circumstances including objects with various properties. In some aspects, determining anticipatory Instruction Sets **526** for autonomous Avatar **605** operation using Collection of Sequences **530*c*** may include selecting a Sequence **533** of Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof from Collection of Sequences **530*c***. Individual and/or collective Similarity Comparisons **125** can be used to determine substantial similarity of the individually and/or collectively compared Collections of Object Representations **525** or portions thereof.

For example, Decision-making Unit **540** can perform Similarity Comparisons **125** of Collections of Object Representations **525*a*1**-**525*an*** or portions thereof from Object Processing Unit **140** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in one or more Sequences **533** of Collection of Sequences **530*c***. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800*ca*** in Sequence **533*wc*** may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Avatar **605** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525*a*1**-**525*an*** and **525*b*1**-**525*bn*** or portions thereof from Object Processing Unit **140** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Sequences **533** of Collection of Sequences **530*c***. Collections of Object Representations **525** or portions thereof from Knowledge Cells **800*ca***-**800*cb*** in Sequence **533*wc*** may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Avatar **605** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525*a*1**-**525*an***, **525*b*1**-**525*bn***, and **525*c*1**-**525*cn*** or portions thereof from Object Processing Unit **140** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Sequences

533 of Collection of Sequences **530*c***. Collections of Object Representations **525** or portions thereof from Knowledge Cells **800*da***-**800*dc*** in Sequence **533*wd*** may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Avatar **605** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525*a*1**-**525*an***, **525*b*1**-**525*bn***, **525*c*1**-**525*cn***, and **525*d*1**-**525*dn*** or portions thereof from Object Processing Unit **140** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Sequences **533** of Collection of Sequences **530*c***. Collections of Object Representations **525** or portions thereof from Knowledge Cells **800*da***-**800*dd*** in Sequence **533*wd*** may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Avatar **605** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525*a*1**-**525*an***, **525*b*1**-**525*bn***, **525*c*1**-**525*cn***, **525*d*1**-**525*dn***, and **525*e*1**-**525*en*** or portions thereof from Object Processing Unit **140** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Sequences **533** of Collection of Sequences **530*c***. Collections of Object Representations **525** or portions thereof from Knowledge Cells **800*da***-**800*de*** in Sequence **533*wd*** may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Avatar **605** operation. Decision-making Unit **540** can implement similar logic or process for any additional Collections of Object Representations **525** from Object Processing Unit **140**, and so on.

The foregoing exemplary embodiment provides an example of utilizing a combination of collective Similarity Comparisons **125**, individual Similarity Comparisons **125**, and/or other elements or techniques. It should be understood that any of these elements and/or techniques can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Sequence **533** of Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info **527** in Similarity Comparisons **125**, traversing of Knowledge Cells **800** or other elements, using history of Collections of Object Representations **525** or Knowledge Cells **800** in collective Similarity Comparisons **125**, using various arrangements of Collections of Object Representations **525** and/or other elements in a Knowledge Cell **800**, and/or others can similarly be utilized in this example. These elements and/or techniques can similarly be utilized in Neural Network **530*a***, Graph **530*b***, Collection of Knowledge Cells **530*d***, and/or other data structures or arrange-

ments. In some aspects, instead of anticipating Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525**, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially matching streams of Collections of Object Representations **525**. In other aspects, any time that substantial similarity or other similarity threshold is not achieved in compared Collections of Object Representations **525** or portions thereof from any of the Knowledge Cells **800**, Decision-making Unit **540** can decide to look for a substantially or otherwise similar Collections of Object Representations **525** or portions thereof in Knowledge Cells **800** elsewhere in Collection of Sequences **530***c* such as in different Sequences **533**. It should be noted that any of Collections of Object Representations **525***a***1**-**525***an*, **525***b***1**-**525***bn*, **525***c***1**-**525***cn*, **525***d***1**-**525***dn*, **525***e***1**-**525***en*, etc. may include one Collection of Object Representations **525** or a stream of Collections of Object Representations **525**. It should also be noted that any Knowledge Cell **800** may include one Collection of Object Representations **525** or a stream of Collections of Object Representations **525** as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring now to Modification Interface **130**. Modification Interface **130** comprises the functionality for modifying execution and/or functionality of Avatar **605**, Application Program **18**, Processor **11**, and/or other processing element. Modification Interface **130** comprises the functionality for modifying execution and/or functionality of Avatar **605**, Application Program **18**, Processor **11**, and/or other processing element at runtime. Modification Interface **130** comprises the functionality for modifying execution and/or functionality of Avatar **605**, Application Program **18**, Processor **11**, and/or other processing element based on anticipatory Instruction Sets **526**. In one example, Artificial Intelligence Unit **110** may determine anticipatory Instruction Sets **526** to be used or executed in Avatar's **605** autonomous operation, and Modification Interface **130** may use the anticipatory Instruction Sets **526** to modify Avatar **605** to effect Avatar's **605** autonomous operation. In another example, in Application Programs **18** that do not comprise Avatar **605** or do not rely on Avatar **605** for their operation, Artificial Intelligence Unit **110** may determine anticipatory Instruction Sets **526** to be used or executed in Application Program's **18** autonomous operation, and Modification Interface **130** may use the anticipatory Instruction Sets **526** to modify Application Program **18** to effect Application Program's **18** autonomous operation. In some aspects, Modification Interface **130** can access, modify, and/or otherwise manipulate runtime engine/environment, virtual machine, operating system, compiler, interpreter, translator, execution stack, file, object, data structure, and/or other computing system elements. In other aspects, Modification Interface **130** can access, modify, and/or otherwise manipulate memory, storage, and/or other repositories. In further aspects, Modification Interface **130** can access, modify, and/or otherwise manipulate Processor **11** registers and/or other Processor **11** elements. In further aspects, Modification Interface **130** can access, modify, and/or otherwise manipulate inputs and/or outputs of Avatar **605**, Application Program **18**, Processor **11**, and/or other processing element. In further aspects, Modification Interface **130** can access, create, delete, modify, and/or otherwise manipulate functions, methods, procedures, routines, subroutines, and/or other elements of Avatar **605** and/or Application Program **18**. In further aspects, Modification Interface **130** can access, create, delete, modify, and/or otherwise manipulate source code, bytecode, compiled code, interpreted code, translated code, machine code, and/or other code. In further aspects, Modification Interface **130** can access, create, delete, modify, and/or otherwise manipulate values, variables, parameters, and/or other data or information. Modification Interface **130** comprises any features, functionalities, and embodiments of Acquisition Interface **120**, and vice versa, as applicable. Modification Interface **130** also comprises other disclosed functionalities.

Modification Interface **130** can employ various techniques for modifying execution and/or functionality of Avatar **605**, Application Program **18**, Processor **11**, and/or other processing element. In some aspects, some of the previously described techniques and/or tools can be utilized. Code instrumentation, for instance, may involve inserting additional code, overwriting or rewriting existing code, and/or branching to a separate segment of code from Application Program **18** as previously described. For example, instrumented code may include the following:

```
Statement1;
Statement2;
modifyAvatar( );
Statement3;
Statement4;
```

In the above sample code, instrumented call to Modification Interface's **130** function such as modifyAvatar( ), modifyApplication( ), etc. can be placed before or after any statement of Avatar **605** and/or Application Program **18** such as after Statement2. A similar call to Modification Interface's **130** function that modifies Avatar **605** and/or Application Program **18** can be placed before or after some or all functions/routines/subroutines, some or all lines of code, some or all statements, some or all instructions or instruction sets, some or all basic blocks, and/or some or all other code segments of Avatar **605** and/or Application Program **18**. One or more calls to functions that modify Avatar **605** and/or Application Program **18** can be placed anywhere in Avatar's **605** and/or Application Program's **18** code and can be executed at any points in Avatar's **605** and/or Application Program's **18** execution. A function that modifies Avatar **605** and/or Application Program **18** may include Artificial Intelligence Unit **110**-determined anticipatory Instruction Sets **526** to be used or executed in Avatar's **605** and/or Application Program's **18** autonomous operation. In some embodiments, the previously described obtaining Avatar's **605** and/or Application Program's **18** instruction sets, data, and/or other information as well as modifying execution and/or functionality of Avatar **605** and/or Application Program **18** can be implemented in a single function that performs both tasks such as traceAndModifyAvatar( ), traceAndModifyApplication( ), etc.

In some embodiments, various computing systems and/or platforms may provide native tools for modifying execution and/or functionality of Avatar **605**, Application Program **18**, Processor **11**, and/or other processing element. Independent vendors may provide tools with similar functionalities that can be utilized across different platforms. These tools enable a wide range of techniques or capabilities such as instrumentation, self-modifying code capabilities, dynamic code capabilities, branching, code rewriting, code overwriting, hot swapping, accessing and/or modifying objects or data structures, accessing and/or modifying functions/routines/subroutines, accessing and/or modifying variable or param-

eter values, accessing and/or modifying processor registers, accessing and/or modifying inputs and/or outputs, providing runtime memory access, and/or other capabilities. One of ordinary skill in art will understand that, while all possible variations of the techniques for modifying execution and/or functionality of Avatar **605**, Application Program **18**, Processor **11**, and/or other processing element are too voluminous to describe, these techniques are within the scope of this disclosure.

In one example, modifying execution and/or functionality of Avatar **605** and/or Application Program **18** can be implemented through utilizing metaprogramming techniques, which include applications that can self-modify or that can create, modify, and/or manipulate other applications. Self-modifying code, dynamic code, reflection, and/or other techniques can be used to facilitate metaprogramming. In some aspects, metaprogramming is facilitated through a programming language's ability to access and manipulate the internals of the runtime engine directly or via an API. In other aspects, metaprogramming is facilitated through dynamic execution of expressions (i.e. anticipatory Instruction Sets **526**, etc.) that can be created and/or executed at runtime. In yet other aspects, metaprogramming is facilitated through application modification tools, which can perform modifications on an application regardless of whether the application's programming language enables metaprogramming capabilities. Some operating systems may protect an application loaded into memory by restricting access to the loaded application. This protection mechanism can be circumvented by utilizing operating system's, processor's, and/or other low level features or commands to unprotect the loaded application. For example, a self-modifying application may modify the in-memory image of itself. To do so, the application can obtain the in-memory address of its code. The application may then change the operating system's or platform's protection on this memory range allowing it to modify the code (i.e. insert anticipatory Instruction Sets **526**, etc.). In addition to a self-modifying application, one application can utilize similar technique to modify another application. Linux mprotect command or similar commands of other operating systems can be used to change protection (i.e. unprotect, etc.) for a region of memory, for example. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Avatar **605** and/or Application Program **18** can be implemented through native capabilities of dynamic, interpreted, and/or scripting programming languages and/or platforms. Most of these languages and/or platforms can perform functionalities at runtime that static programming languages may perform during compilation. Dynamic, interpreted, and/or scripting languages provide native functionalities such as self-modification of code, dynamic code, extending the application, adding new code, extending objects and definitions, and/or other functionalities that can modify an application's execution and/or functionality at runtime. Examples of dynamic, interpreted, and/or scripting languages include Lisp, Perl, PHP, JavaScript, Ruby, Python, Smalltalk, Td, VBScript, and/or others. Similar functionalities can also be provided in languages such as Java, C, and/or others using reflection. Reflection includes the ability of an application to examine and modify the structure and behavior of the application at runtime. For example, JavaScript can modify its own code as it runs by utilizing Function object constructor as follows:

```
myFunc=new Function(arg1, arg2, argN, functionBody);
```

The sample code above causes a new function object to be created with the specified arguments and body. The body and/or arguments of the new function object may include new instruction sets (i.e. anticipatory Instruction Sets **526**, etc.). The new function can be invoked as any other function in the original code. In another example, JavaScript can utilize eval method that accepts a string of JavaScript statements (i.e. anticipatory Instruction Sets **526**, etc.) and execute them as if they were within the original code. An example of how eval method can be used to modify an application includes the following JavaScript code:

```
anticipatoryInstr='Avatar.moveForward(14);';
if (anticipatoryInstr!=" " && anticipatoryInstr!=null)
{
  eval(anticipatoryInstr);
}
```

In the sample code above, Artificial Intelligence Unit **110** may generate anticipatory Instruction Set **526** (i.e. 'Avatar.moveForward(14)' for moving Avatar **605** forward 14 units, etc.) and save it in anticipatoryInstr variable, which eval method can then execute. Lisp is another example of dynamic, interpreted, and/or scripting language that includes similar capabilities as the aforementioned JavaScript. For example, Lisp's compile command can create a function at runtime, eval command may parse and evaluate an expression at runtime, and exec command may execute a given instruction set (i.e. string, etc.) at runtime. In another example, dynamic as well as some non-dynamic languages may provide macros, which combine code introspection and/or eval capabilities. In some aspects, macros can access inner workings of the compiler, interpreter, virtual machine, runtime environment/engine, and/or other components of the computing platform enabling the definition of language-like constructs and/or generation of a complete program or sections thereof. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Avatar **605** and/or Application Program **18** can be implemented through dynamic code, dynamic class loading, reflection, and/or other functionalities of a programming language or platform. In static applications or static programming, a class can be defined and/or loaded at compile time. Conversely, in dynamic applications or dynamic programming, a class can be loaded into a running environment at runtime. For example, Java Runtime Environment (JRE) may not require that all classes be loaded at compile time and class loading can occur when a class is first referenced at runtime. Dynamic class loading enables inclusion or injection of on-demand code and/or functionalities at runtime. System provided or custom class loaders may enable loading of classes into the running environment. Custom class loaders can be created to enable custom functionalities such as, for example, specifying a remote location from which a class can be loaded. In addition to dynamic loading of a pre-defined class, a class can also be created at runtime. In some aspects, a class source code can be created at runtime. A compiler such as javac, com.sun.tools.javac.Main, javax.tools, javax.tools.JavaCompiler, and/or other packages can then be utilized to compile the source code. Javac, com.sun.tools.javac.Main, javax.tools, javax.tools.JavaCompiler, and/or other packages may include an interface to invoke Java compiler from within a running application. A Java compiler may accept source code in a file, string, object (i.e. Java String, StringBuffer, CharSequence, etc.) and/or other source, and may generate Java bytecode (i.e. class file, etc.). Once compiled, a class loader can then load the compiled class into the running environment. In other aspects, a tool such as Javaassist (i.e. Java programming

assistant) can be utilized to enable an application to create or modify a class at runtime. Javassist may include a Java library that provides functionalities to create and/or manipulate Java bytecode of an application as well as reflection capabilities. Javassist may provide source-level and bytecode-level APIs. Using the source-level API, a class can be created and/or modified using only source code, which Javassist may compile seamlessly on the fly. Javassist source-level API can therefore be used without knowledge of Java bytecode specification. Bytecode-level API enables creating and/or editing a class bytecode directly. In yet other aspects, similar functionalities to the aforementioned ones may be provided in tools such as Apache Commons BCEL (Byte Code Engineering Library), ObjectWeb ASM, CGLIB (Byte Code Generation Library), and/or others. Once a dynamic code or class is created and loaded, reflection in high-level programming languages such as Java and/or others can be used to manipulate or change the runtime behavior of an application. Examples of reflective programming languages and/or platforms include Java, JavaScript, Smalltalk, Lisp, Python, .NET Common Language Runtime (CLR), Td, Ruby, Pert, PHP, Scheme, PL/SQL, and/or others. Reflection can be used in an application to access, examine, modify, and/or manipulate a loaded class and/or its elements. Reflection in Java can be implemented by utilizing a reflection API such as java.lang.Reflect package. The reflection API provides functionalities such as, for example, loading or reloading a class, instantiating a new instance of a class, determining class and instance methods, invoking class and instance methods, accessing and manipulating a class, fields, methods and constructors, determining the modifiers for fields, methods, classes, and interfaces, and/or other functionalities. The above described dynamic code, dynamic class loading, reflection, and/or other functionalities are similarly provided in the .NET platform through its tools such as, for example, System.CodeDom.Compiler namespace, System.Reflection.Emit namespace, and/or other native or other .NET tools. Other platforms in addition to Java and .NET may provide similar tools and/or functionalities. In some designs, dynamic code, dynamic class loading, reflection, and/or other functionalities can be used to facilitate modification of an application by inserting or injecting instruction sets (i.e. anticipatory Instruction Sets **526**, etc.) into a running application. For example, an existing or dynamically created class comprising ACAAO Unit **100** functionalities can be loaded into a running application through manual, automatic, or dynamic instrumentation. Once the class is created and loaded, an instance of ACAAO Unit **100** class may be constructed. The instance of ACAAO Unit **100** can then take or exert control of the application and/or implement alternate instruction sets (i.e. anticipatory Instruction Sets **526**, etc.) at any point in the application's execution. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Avatar **605** and/or Application Program **18** can be implemented through independent tools that can be utilized across different platforms. Such tools provide instrumentation and/or other capabilities on more than one platform or computing system and may facilitate application modification or insertion of instruction sets (i.e. anticipatory Instruction Sets **526**, etc.). Examples of these tools include Pin, DynamoRIO, DynInst, Kprobes, KemInst, OpenPAT, DTrace, SystemTap, and/or others. In some aspects, Pin and/or any of its elements, methods, and/or techniques can be utilized for dynamic instrumentation. Pin can perform

instrumentation by taking control of an application after it loads into memory. Pin may insert itself into the address space of an executing application enabling it to take control. Pin JIT compiler can then compile and implement alternate code (i.e. anticipatory Instruction Sets **526**, etc.). Pin provides an extensive API for instrumentation at several abstraction levels. Pin supports two modes of instrumentation, JIT mode and probe mode. JIT mode uses a just-in-time compiler to insert instrumentation and recompile program code while probe mode uses code trampolines for instrumentation. Pin was designed for architecture and operating system independence. In other aspects, KemInst and/or any of its elements, methods, and/or techniques can be utilized for dynamic instrumentation. KemInst includes an instrumentation framework designed for dynamically inserting code into a running kernel of an operating system. KemInst implements probe-based dynamic instrumentation where code can be inserted, changed, and/or removed at will. Keminst API enables client tools to construct their own tools for dynamic kernel instrumentation to suit variety of purposes such as insertion of alternate instruction sets (i.e. anticipatory Instruction Sets **526**, etc.). Client tools can communicate with KemInst over a network (i.e. internet, wireless network, LAW, WAN, etc). Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Avatar **605** and/or Application Program **18** can be implemented through utilizing operating system's native tools or capabilities such as Unix ptrace command. Ptrace includes a system call that may enable one process to control another allowing the controller to inspect and manipulate the internal state of its target. Ptrace can be used to modify a running application such as modify an application with alternate instruction sets (i.e. anticipatory Instruction Sets **526**, etc.). By attaching to an application using the ptrace call, the controlling application can gain extensive control over the operation of its target. This may include manipulation of its instruction sets, execution path, file descriptors, memory, registers, and/or other components. Ptrace can single-step through the targets code, observe and intercept system calls and their results, manipulate the target's signal handlers, receive and send signals on the target's behalf, and/or perform other operations within the target application. Ptrace's ability to write into the target application's memory space enables the controller to modify the running code of the target application. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Avatar **605** and/or Application Program **18** can be implemented through utilizing just-in-time (JIT) compiling. JIT compilation (also known as dynamic translation, dynamic compilation, etc.) includes compilation performed during an application's execution (i.e. runtime, etc.). A code can be compiled when it is about to be executed, and it may be cached and reused later without the need for additional compilation. In some aspects, a JIT compiler can convert source code or byte code into machine code. In other aspects, a JIT compiler can convert source code into byte code. JIT compiling may be performed directly in memory. For example, JIT compiler can output machine code directly into memory and immediately execute it. Platforms such as Java, .NET, and/or others may implement JIT compilation as their native functionality. Platform independent tools for custom system design may include JIT compilation functionalities as well. In some aspects, JIT compilation includes

redirecting application's execution to a JIT compiler from a specific entry point. For example, Pin can insert its JIT compiler into the address space of an application. Once execution is redirected to it, JIT compiler may receive alternate instruction sets (i.e. anticipatory Instruction Sets **526**, etc.) immediately before their compilation. The JIT compiled instruction sets may be stored in memory or another repository from where they can be retrieved and executed. Alternatively, for example, JIT compiler can create a copy of the original application code or a segment thereof and insert alternate code (i.e. anticipatory Instruction Sets **526**, etc.) before compiling the modified code copy. In some aspects, JIT compiler may include a specialized memory such as fast cache memory dedicated to JIT compiler functionalities from which the modified code can be fetched rapidly. JIT compilation and/or any compilation in general may include compilation, interpretation, or other translation into machine code, bytecode, and/or other formats or types of code. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Avatar **605** and/or Application Program **18** can be implemented through dynamic recompilation. Dynamic recompilation includes recompiling an application or part thereof during execution. An application can be modified with alternate features or instruction sets that may take effect after recompilation. Dynamic recompilation may be practical in various types of applications including object oriented, event driven, forms based, and/or other applications. In a typical windows-based application, most of the action after initial startup occurs in response to user or system events such as moving the mouse, selecting a menu option, typing text, running a scheduled task, making a network connection, and/or other events when an event handler is called to perform an operation appropriate for the event. Generally, when no events are being generated, the application is idle. For example, when an event occurs and an appropriate event handler is called, instrumentation can be implemented in the application's source code to insert alternate instruction sets (i.e. anticipatory Instruction Sets **526**, etc.) at which point the modified source code can be recompiled and/or executed. In some aspects, the state of the application can be saved before recompiling its modified source code so that the application may continue from its prior state. Saving the application's state can be achieved by saving its variables, data structures, objects, location of its current instruction, and/or other necessary information in environmental variables, memory, or other repositories where they can be accessed once the application is recompiled. In other aspects, application's variables, data structures, objects, address of its current instruction, and/or other necessary information can be saved in a repository such as file, database, or other repository accessible to the application after recompilation of its source code. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Avatar **605** and/or Application Program **18** can be implemented through modifying or redirecting application's execution path. Generally, an application can be loaded into memory and the flow of execution proceeds from one instruction set to the next until the end of the application. An application may include a branching mechanism that can be driven by keyboard or other input devices, system events, and/or other computing system components or events that may impact the execution path. The execution

path can also be altered by an external application through acquiring control of execution and/or redirecting execution to a function, routine/subroutine, or an alternate code segment at any point in the application's execution. A branch, jump, or other mechanism can be utilized to implement the redirected execution. For example, a jump instruction can be inserted at a specific point in an application's execution to redirect execution to an alternate code segment. A jump instruction set may include, for example, an unconditional branch, which always results in branching, or a conditional branch, which may or may not result in branching depending on a condition. When executing an application, a computer may fetch and execute instruction sets in sequence until it encounters a branch instruction set. If the instruction set is an unconditional branch, or it is conditional and the condition is satisfied, the computer may fetch its next instruction set from a different instruction set sequence or code segment as specified by the branch instruction set. After the execution of the alternate code segment, control may be redirected back to the original jump point or to another point in the application. For example, modifying an application can be implemented by redirecting execution of an application to alternate instruction sets (i.e. anticipatory Instruction Sets **526**, etc.). Alternate instruction sets can be pre-compiled, pre-interpreted, or otherwise pre-translated and ready for execution. Alternate instruction sets can also be JIT compiled, JIT interpreted, or otherwise JIT translated before execution. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Avatar **605** and/or Application Program **18** can be implemented through assembly language. Assembly language instructions may be directly related with the architecture's machine instructions as previously described. Assembly language can, therefore, be a powerful tool for implementing direct hardware (i.e. processor registers, memory, etc.) access and manipulations as well as access and manipulations of specialized processor features or instructions. Assembly language can also be a powerful tool for implementing low-level embedded systems, real-time systems, interrupt handlers, self or dynamically modifying code, and/or other applications. Specifically, for instance, self or dynamically modifying code that can be used to facilitate modifying of an application can be seamlessly implemented using assembly language. For example, using assembly language, instruction sets can be dynamically created and loaded into memory similar to the ones that a compiler may generate. Furthermore, using assembly language, memory space of a loaded application can be accessed to modify (including rewrite, overwrite, etc.) original instruction sets or to insert jumps or branches to alternate code elsewhere in memory. Some operating systems may implement protection from changes to applications loaded into memory. Operating system's, processor's, or other low level features or commands can be used to unprotect the protected locations in memory before the change as previously described. Alternatively, a pointer that may reside in a memory location where it could be readily altered can be utilized where the pointer may reference alternate code. In one example, assembly language can be utilized to write alternate code (i.e. anticipatory Instruction Sets **526**, etc.) into a location in memory outside a running application's memory space. Assembly language can then be utilized to redirect the application's execution to the alternate code by inserting a jump or branch into the application's in-memory code, by redirecting program counter, or by other technique.

In another example, assembly language can be utilized to overwrite or rewrite the entire or part of an application's in-memory code with alternate code. In some aspects, high-level programming languages can cal an external assembly language program to facilitate application modification as previously described. In yet other aspects, relatively low-level programming languages such as C may allow embedding assembly language directly in their source code such as, for example, using asm keyword of C. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Avatar **605** and/or Application Program **18** can be implemented through binary rewriting. Binary rewriting tools and/or techniques may modify an application's executable. In some aspects, modification can be minor such as in the case of optimization where the original executable's functionality is kept. In other aspects, modification may change the application's functionality such as by inserting alternate code (i.e. anticipatory Instruction Sets **526**, etc.). Examples of binary rewriting tools include SecondWrite, ATOM, DynamoRIO, Purify, Pin, EEL, DynInst, PLTO, and/or others. Binary rewriting may include disassembly, analysis, and/or modification of target application. Since binary rewriting works directly on machine code executable, it is independent of source language, compiler, virtual machine (if one is utilized), and/or other higher level abstraction layers. Also, binary rewriting tools can perform application modifications without access to original source code. Binary rewriting tools include static rewriters, dynamic rewriters, minimally-invasive rewriters, and/or others. Static binary rewriters can modify an executable when the executable is not in use (i.e. not running). The rewritten executable may then be executed including any new or modified functionality. Dynamic binary rewriters can modify an executable during its execution, thereby enabling modification of an application's functionality at runtime. In some aspects, dynamic rewriters can be used for instrumentation or selective modifications such as insertion of alternate code (i.e. anticipatory Instruction Sets **526**, etc.), and/or for other runtime transformations or modifications. For example, some dynamic rewriters can be configured to intercept an application's execution at indirect control transfers and insert instrumentation or other application modifying code. Minimally-invasive rewriters may keep the original machine code to the greatest extent possible. They support limited modifications such as insertion of jumps into and out of instrumented code. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

Referring to FIG. **29**, in a further example, modifying execution and/or functionality of Avatar **605** and/or Application Program **18** can be implemented through modification of processor registers, memory, or other computing system components. In some aspects, modifying execution and/or functionality of Processor **11** can be implemented by redirecting Processor's **11** execution to alternate instruction sets (i.e. anticipatory Instruction Sets **526**, etc.). In one example, Program Counter **211** may hold or point to a memory address of the next instruction set that will be executed by Processor **11**. Artificial Intelligence Unit **110** may generate anticipatory Instruction Sets **526** and store them in Memory **12** as previously described. Modification Interface **130** may then change Program Counter **211** to point to the location in Memory **12** where anticipatory Instruction Sets **526** are stored. The anticipatory Instruction Sets **526** can then be fetched from the location in Memory **12** pointed to by the modified Program Counter **211** and loaded into Instruction Register **212** for decoding and execution. Once anticipatory Instruction Sets **526** are executed, Modification Interface **130** may change Program Counter **211** to point to the last instruction set before the redirection or to any other instruction set. In other aspects, anticipatory Instruction Sets **526** can be loaded directly into Instruction Register **212**. As previously described, examples of other processor or computing system components that can be used during an instruction cycle include memory address register (MAR), memory data register (MDR), data registers, address registers, general purpose registers (GPRs), conditional registers, floating point registers (FPRs), constant registers, special purpose registers, machine-specific registers, Register Array **214**, Arithmetic Logic Unit **215**, control unit, and/or other circuits or components. Any of the aforementioned processor registers, memory, or other computing system components can be accessed and/or modified to facilitate the disclosed functionalities. In some embodiments, processor interrupt may be issued to facilitate such access and/or modification. In some designs, modifying execution and/or functionality of Processor **11** can be implemented in a program, combination of programs and hardware, or purely hardware system. Dedicated hardware may be built to perform modifying execution and/or functionality of Processor **11** with marginal or no impact to computing overhead. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

Other additional techniques or elements can be utilized as needed for modifying execution and/or functionality of Avatar **605**, Application Program **18**, Processor **11**, and/or other processing elements, or some of the disclosed techniques or elements can be excluded, or a combination thereof can be utilized in alternate embodiments. As an avatar (i.e. Avatar **605**, etc.) may be part of an application (i.e. Application Program **18**, etc.), it should be noted that modifying execution and/or functionality of an avatar may include same or similar techniques as the aforementioned modifying execution and/or functionality of an application, and vice versa.

Referring to FIG. **30**, the illustration shows an embodiment of a method **9100** for learning and/or using an avatar's circumstances for autonomous avatar operation. In some aspects, the method can be used on a computing device or system to enable learning of an avatar's operation in circumstances including objects with various properties and enable autonomous avatar operation in similar circumstances. Method **9100** may include any action or operation of any of the disclosed methods such as method **9200**, **9300**, **9400**, **9500**, **9600**, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method **9100**.

At step **9105**, a first collection of object representations is received. A collection of object representations (i.e. Collection of Object Representations **525**, etc.) may include one or more object representations (i.e. Object Representations **625**, etc.), object properties (i.e. Object Properties **630**, etc.), and/or other elements or information. In some designs, an object representation includes a representation of an object (i.e. Object **615**, etc.) in an avatar's (i.e. Avatar's **605**, etc.) surrounding within an application (i.e. Application Program **18**, etc.). In other designs, an object representation includes a representation of an object of an application. As such, an object representation may include any information related to an object. In further designs, an object representation may

include or be replaced with an object itself, in which case the object representation as an element can be optionally omitted. In some embodiments, a collection of object representations may include one or more object representations, object properties, and/or other elements or information obtained in an avatar's surrounding at a particular time. A collection of object representations may, therefore, include knowledge (i.e. unit of knowledge, etc.) of an avatar's circumstances including objects with various properties at a particular time. In other embodiments, where an application does not comprise an avatar or rely on an avatar for its operation, a collection of object representations may include one or more object representations, object properties, and/or other elements or information obtained in an application or part thereof at a particular time. A collection of object representations may, therefore, include knowledge (i.e. unit of knowledge, etc.) of an application's circumstances including objects with various properties at a particular time. In some designs, a collection of object representations may include or be associated with a time stamp (not shown), order (not shown), or other time related information. In further embodiments, a collection of object representations may include or be substituted with a stream of collections of object representations, and vice versa. Therefore, the terms collection of object representations and stream of collections of object representations may be used interchangeably depending on context. A stream of collections of object representations may include one collection of object representations or a group, sequence, or other plurality of collections of object representations. In some aspects, a stream of collections of object representations may include one or more collections of object representations, and/or other elements or information obtained in an avatar's surrounding over time. A stream of collections of object representations may, therefore, include knowledge (i.e. unit of knowledge, etc.) of an avatar's circumstances including objects with various properties over time. As circumstances including objects with various properties in an avatar's surrounding change (i.e. objects and/or their properties change, move, act, transform, etc.) over time, this change may be captured in a stream of collections of object representations. In further embodiments, where an application does not comprise an avatar or rely on an avatar for its operation, a stream of collections of object representations may include one or more collections of object representations, and/or other elements or information obtained in an application or part thereof over time. A stream of collections of object representations may, therefore, include knowledge (i.e. unit of knowledge, etc.) of an application's circumstances including objects with various properties over time. In some designs, each collection of object representations in a stream may include or be associated with the aforementioned time stamp, order, or other time related information. Examples of objects include models of a person, animal, tree, rock, building, vehicle, and/or others in a context of a computer game, virtual world, 3D or 2D graphics application, and/or others. More generally, examples of objects include a 2D model, a 3D model, a 2D shape (i.e. point, line, square, rectangle, circle, triangle, etc.), a 3D shape (i.e. cube, sphere, etc.), a graphical user interface (GUI) element, a form element (i.e. text field, radio button, push button, check box, etc.), a data or database element, a spreadsheet element, a link, a picture, a text (i.e. character, word, etc.), a number, and/or others in a context of a web browser, a media application, a word processing application, a spreadsheet application, a database application, a forms-based application, an operating system, a device/system control application, and/or others. In some aspects, any part of an object can be identified as an object itself. For instance, instead of or in addition to identifying a building as an object, a window, door, roof, and/or other parts of the building can be identified as objects. In general, an object may include any object or part thereof that can be obtained or recognized. Examples of object properties include existence of an object, type of an object (i.e. person, animal, tree, rock, building, vehicle, etc.), identity of an object (i.e. name, identifier, etc.), distance of an object, bearing/angle of an object, location of an object (i.e. distance and bearing/angle from a known point, coordinates, etc.), shape/size of an object (i.e. scale, height, width, depth, computer model, etc.), activity of an object (i.e. motion, gestures, etc.), and/or other properties of an object. In general, an object property may include any attribute of an object (i.e. existence of an object, type of an object, identity of an object, shape/size of an object, etc.), any relationship of an object with an avatar, other objects, or the environment (i.e. distance of an object, bearing/angle of an object, friend/foe relationship, etc.), and/or other information related to an object. In some designs, objects and/or their properties can be obtained from an engine, environment, or other system used to implement an application. For instance, objects and/or their properties can be obtained by utilizing functions for providing properties or other information on objects of an engine, environment, or other system used to implement an application. Examples of such engines, environments, or other systems include Unity 3D Engine, Unreal Engine, Torque 3D Engine, and/or others. In other designs, objects and/or their properties can be obtained by accessing and/or reading a scene graph or other data structure used for organizing objects in a particular application, or in an engine, environment, or other system used to implement an application. In other designs, objects and/or their properties can be detected or recognized from one or more pictures depicting views of an avatar's surrounding or views of an application. Any picture recognition techniques (i.e. Picture Recognizer **92**, etc.) can be used for such detection or recognizing. The one or more pictures depicting views of an avatar's surrounding or views of an application can be rendered or generated by a picture renderer (i.e. Picture Renderer **91**, etc.). In further designs, objects and/or their properties can be detected or recognized from one or more sounds from an avatar's surrounding or one or more sounds of an application. Any sound recognition techniques (i.e. Sound Recognizer **97**, etc.) can be used for such detection or recognizing. The one or more sounds from an avatar's surrounding or one or more sounds of an application can be rendered or generated by a sound renderer (i.e. Sound Renderer **96**, etc.). Receiving comprises any action or operation by or for an Object **615**, Collection of Object Representations **525**, stream of Collections of Object Representations **525**, Object Representation **625**, Object Property **630**, Object Processing Unit **140**, Picture Renderer **91**, Sound Renderer **96**, Picture Recognizer **92**, Sound Recognizer **97**, and/or other disclosed elements.

At step **9110**, a first one or more instruction sets for operating an avatar are received. In some aspects, an instruction set (i.e. Instruction Set **526**, etc.) may be used or executed for operating an avatar (i.e. Avatar **605**, etc.) of an application (i.e. Application Program **18**, etc.). In other aspects, where an application does not comprise an avatar or rely on an avatar for its operation, an instruction set may be used or executed for operating the application. Therefore, a reference to an instruction set for operating an avatar includes or can be substituted with a reference to an instruction set for operating an application depending on context.

In some embodiments, an instruction set can be used or executed by a processor (i.e. Processor **11**, etc.) in operating an avatar and/or application. Operating an avatar and/or application includes performing or causing any operations on/by/with the avatar and/or application. In some designs, an instruction set can be received from an avatar, application, processor, and/or other processing element as the instruction set is being used or executed. In other designs, an instruction set can be received from an avatar, application, processor, and/or other processing element before or after the instruction set is used or executed. In further designs, an instruction set can be received from a running avatar, running application, running processor, and/or other running processing element. As such, an instruction set can be received at runtime. In some embodiments, receiving an instruction set includes tracing or profiling an avatar, application, processor, and/other processing elements. Tracing or profiling may include adding trace code or instrumentation to an avatar (i.e. avatar's object code, etc.) or application, and/or outputting trace information (i.e. instruction sets, etc.) to a receiving target. In some aspects, instrumentation can be performed in source code, bytecode, compiled code, interpreted code, translated code, machine code, and/or other code. In other aspects, instrumentation can be performed in various elements of a computing system such as memory, virtual machine, runtime engine/environment, operating system, compiler, interpreter, translator, processor registers, and/or other elements. In yet other aspects, instrumentation can be performed in various abstraction layers of a computing system such as in software layer (i.e. application, etc.), in virtual machine (if VM is used), in operating system, in processor, and/or in other layers or areas that may exist in a particular computing system implementation. In yet other aspects, instrumentation can be performed at various times in an avatar's or application's execution such as source code write time, compile time, interpretation time, translation time, linking time, loading time, runtime, and/or others. In yet other aspects, instrumentation can be performed at various granularities or code segments such as some or all lines of code, some or all statements, some or all instructions or instruction sets, some or all basic blocks, some or all functions/routines/subroutines, and/or some or all other code segments. In yet other aspects, instrumentation may include a manual, automatic, dynamic, or just-in-time (JIT) instrumentation. In general, any instrumentation technique can be utilized. In further embodiments, receiving an instruction set includes attaching to or interfacing with an avatar, application, processor, and/or other processing element. In further embodiments, receiving an instruction set includes accessing and/or reading runtime engine/environment, virtual machine, operating system, compiler, interpreter, translator, execution stack, file, object, data structure, and/or other computing system elements. In further embodiments, receiving an instruction set includes accessing and/or reading memory, storage, and/or other repository. In further embodiments, receiving an instruction set includes accessing and/or reading processor registers and/or other processor elements. In further embodiments, receiving an instruction set includes accessing and/or reading inputs and/or outputs of an avatar, application, processor, and/or other processing element. In further embodiments, receiving an instruction set includes accessing and/or reading functions, methods, procedures, routines, subroutines, and/or other elements of an avatar and/or application. In further embodiments, receiving an instruction set includes accessing and/or reading source code, bytecode, compiled code, interpreted code, translated code, machine code, and/or other code. In further embodiments, receiving an instruction set includes accessing and/or reading values, variables, parameters, and/or other data or information. One or more instruction sets may temporally correspond to a collection of object representations. In general, one or more instruction sets that temporally correspond to a collection of object representations enable structuring knowledge of an avatar's and/or application's operation at or around the time of generating the collection of object representations. Such functionality enables spontaneous or seamless learning of an avatar's and/or application's operation in circumstances including objects with various properties as the avatar and/or application are operated in real time. Receiving comprises any action or operation by or for an Acquisition Interface **120**, Instruction Set **526**, and/or other disclosed elements.

At step **9115**, the first collection of object representations is correlated with the first one or more instruction sets for operating the avatar. In some aspects, individual collections of object representations can be correlated with one or more instruction sets. In other aspects, streams of collections of object representations can be correlated with one or more instruction sets. In further aspects, individual collections of object representations or streams of collections of object representations can be correlated with the aforementioned temporally corresponding instruction sets. In further aspects, a collection of object representations or stream of collections of object representations may not be correlated with any instruction sets. Correlating may include structuring or generating a knowledge cell (i.e. Knowledge Cell **800**, etc.) and storing one or more collections of object representations correlated with any instruction sets into the knowledge cell. Therefore, a knowledge cell may include any data structure or arrangement that can facilitate such storing. A knowledge cell includes knowledge (i.e. unit of knowledge, etc.) of how an avatar and/or application operated in a circumstance including objects with various properties. In some designs, extra information (i.e. Extra Info **527**, etc.) can optionally be used to facilitate enhanced comparisons or decision making in autonomous avatar and/or application operation where applicable. Therefore, any collection of object representations, instruction set, and/or other element may include or be correlated with extra information. Extra information may include any information useful in comparisons or decision making performed in autonomous avatar and/or application operation. Examples of extra information include time information, location information, computed information, visual information, acoustic information, contextual information, and/or other information. Correlating can be omitted where learning of an avatar's and/or application's operation in circumstances including objects with various properties is not implemented. Correlating comprises any action or operation by or for a Knowledge Structuring Unit **520**, Knowledge Cell **800**, and/or other disclosed elements.

At step **9120**, the first collection of object representations correlated with the first one or more instruction sets for operating the avatar are stored. A collection of object representations correlated with one or more instruction sets may be part of a stored plurality of collections of object representations correlated with one or more instruction sets. Collections of object representations correlated with any instruction sets can be stored in a memory unit or other repository. The aforementioned knowledge cells comprising collections of object representations correlated with any instruction sets can be used in/as neurons, nodes, vertices, or other elements in any of the data structures or arrangements (i.e. neural network, graph, sequence, collection of knowledge cells, etc.) used for storing the knowledge of an

avatar's and/or application's operation in circumstances including objects with various properties. Knowledge cells may be interconnected, interrelated, or interlinked into knowledge structures using statistical, artificial intelligence, machine learning, and/or other models or techniques. Such interconnected or interrelated knowledge cells can be used for enabling an avatar's and/or application's autonomous operation. The interconnected or interrelated knowledge cells may be stored or organized into a knowledgebase (i.e. Knowledgebase **530**, etc.). In some embodiments, knowledgebase may be or include a neural network (i.e. Neural Network **530***a*, etc.). In other embodiments, knowledgebase may be or include a graph (i.e. Graph **530***b*, etc.). In further embodiments, knowledgebase may be or include a collection of sequences (i.e. Collection of Sequences **530***c*, etc.). In further embodiments, knowledgebase may be or include a sequence (i.e. Sequence **533**, etc.). In further embodiments, knowledgebase may be or include a collection of knowledge cells (i.e. Collection of Knowledge Cells **530***d*, etc.). In general, knowledgebase may be or include any data structure or arrangement, and/or repository capable of storing the knowledge of an avatar's and/or application's operation in circumstances including objects with various properties. Knowledgebase may also include or be substituted with various artificial intelligence methods, systems, and/or models for knowledge structuring, storing, and/or representation such as deep learning, supervised learning, unsupervised learning, neural networks (i.e. convolutional neural network, recurrent neural network, deep neural network, etc.), search-based, logic and/or fuzzy logic-based, optimization-based, tree/graph/other data structure-based, hierarchical, symbolic and/or sub-symbolic, evolutionary, genetic, multi-agent, deterministic, probabilistic, statistical, and/or other methods, systems, and/or models. Storing can be omitted where learning of an avatar's and/or application's operation in circumstances including objects with various properties is not implemented. Storing comprises any action or operation by or for a Memory **12**, Storage **27**, Knowledgebase **530**, Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, Knowledge Cell **800**, Node **852**, Connection **853**, Layer **854**, Similarity Comparison **125**, and/or other disclosed elements.

At step **9125**, a new collection of object representations is received. Step **9125** may include any action or operation described in Step **9105** as applicable.

At step **9130**, the new collection of object representations is compared with the first collection of object representations. Comparing one collection of object representations with another collection of object representations may include comparing at least a portion of one collection of object representations with at least a portion of the other collection of object representations. In some embodiments, collections of object representations may be compared individually. In some aspects, comparing of individual collections of object representations may include comparing one or more object representations of one collection of object representations with one or more object representations of another collection of object representations. In other aspects, comparing of object representations may include comparing one or more object properties of one object representation with one or more object properties of another object representation. In some designs, one or more object properties in the same category (i.e. Category **635**, etc.) can be compared. Comparing may include any techniques for comparing text, numbers, and/or other data. In further aspects, some object representations, object properties, and/or other elements of a collection of object representations can be omitted from comparison depending on implementation. In other embodiments, collections of object representations may be compared collectively as part of streams of collections of object representations. Collective comparing of collections of object representations may include any features, functionalities, and embodiments of the aforementioned individual comparing of collections of object representations. In some aspects, collective comparing of collections of object representations may include comparing one or more collections of object representations of one stream of collections of object representations with one or more collections of object representations of another stream of collections of object representations. In some designs, one or more corresponding (i.e. similarly ordered, temporally related, etc.) collections of object representations from the compared streams of collections of object representations can be compared. In other designs, Dynamic Time Warping (DTW) and/or other techniques can be utilized for comparison and/or aligning temporal sequences (i.e. streams of collections of object representations, etc.) that may vary in time or speed. In some aspects, some collections of object representations can be omitted from comparison depending on implementation. Any combination of the aforementioned and/or other elements or techniques can be utilized in alternate embodiments of the comparing. Comparing can be omitted where anticipating of an avatar's and/or application's operation in circumstances including objects with various properties is not implemented. Comparing comprises any action or operation by or for a Similarity Comparison **125**, Decision-making Unit **540**, and/or other disclosed elements.

At step **9135**, a determination is made that there is at least a partial match between the new collection of object representations and the first collection of object representations. In some embodiments, determining at least a partial match between individually compared collections of object representations includes determining that a similarity between one or more portions of one collection of object representations and one or more portions of another collection of object representations exceeds a similarity threshold. In other embodiments, determining at least a partial match between individually compared collections of object representations includes determining at least a partial match between one or more portions of one collection of object representations and one or more portions of another collection of object representations. In further embodiments, determining at least a partial match between individually compared collections of object representations includes determining substantial similarity between one or more portions of one collection of object representations and one or more portions of another collection of object representations. A portion of a collection of object representations may include an object representation, an object property, and/or other portion or element of the collection of object representations. In further embodiments, determining at least a partial match between individually compared collections of object representations includes determining that the number or percentage of matching or substantially matching object representations of the compared collections of object representations exceeds a threshold number (i.e. 1, 2, 4, 7, 18, etc.) or threshold percentage (i.e. 41%, 62%, 79%, 85%, 93%, etc.). In some aspects, type of object representations, importance of object representations, and/or other elements or techniques relating to object representations can be utilized for determining similarity using object representations. In further aspects, some of the object representations can be omitted in determining similarity using object representa-

tions depending on implementation. In further embodiments, determining a match or substantial match between compared object representations includes determining that the number or percentage of matching or substantially matching object properties of the compared object representations exceeds a threshold number (i.e. 1, 2, 3, 6, 11, etc.) or a threshold percentage (i.e. 55%, 61%, 78%, 82%, 99%, etc.). In some aspects, categories of object properties, importance of object properties, and/or other elements or techniques relating to object properties can be utilized for determining similarity using object properties. In further aspects, some of the object properties can be omitted in determining similarity using object properties depending on implementation. In some designs, substantial similarity of individually compared collections of object representations can be achieved when a similarity between one or more portions of one collection of object representations and one or more portions of another collection of object representations exceeds a similarity threshold. In other designs, substantial similarity of individually compared collections of object representations can be achieved when the number or percentage of matching or substantially matching object representations of the compared collections of object representations exceeds a threshold number (i.e. 1, 2, 4, 7, 18, etc.) or threshold percentage (i.e. 41%, 62%, 79%, 85%, 93%, etc.). In further aspects, substantial similarity of compared object representations can be achieved when the number or percentage of matching or substantially matching object properties of the compared object representations exceeds a threshold number (i.e. 1, 2, 3, 6, 11, etc.) or a threshold percentage (i.e. 55%, 61%, 78%, 82%, 99%, etc.). In some embodiments, determining at least a partial match between collectively compared collections of object representations (i.e. streams of collections of object representations, etc.) includes determining that the number or percentage of matching or substantially matching collections of object representations of the compared streams of collections of object representations exceeds a threshold number (i.e. 1, 2, 4, 9, 33, 138, etc.) or threshold percentage (i.e. 39%, 58%, 77%, 88%, 94%, etc.). In some aspects, importance of collections of object representations, order of collections of object representations, and/or other elements or techniques relating to collections of object representations can be utilized for determining similarity of collectively compared collections of object representations or streams of collections of object representations. In further aspects, some of the collections of object representations can be omitted in determining similarity of collectively compared collections of object representations or streams of collections of object representations depending on implementation. In some designs, a threshold for a number or percentage similarity can be used to determine a match or substantial match between any of the aforementioned elements. Any text, number, and/or other data similarity determination techniques can be used in any of the aforementioned similarity determinations. A partial match of any of the compared elements may include a substantially or otherwise similar match, and vice versa. Therefore, these terms may be used interchangeably herein depending on context. Although, substantial similarity or substantial match is frequently used herein, it should be understood that any level of similarity, however high or low, may be utilized as defined by the rules (i.e. thresholds, etc.) for similarity. Any combination of the aforementioned and/or other elements or techniques can be utilized in alternate embodiments. Determining can be omitted where anticipating of an avatar's and/or application's operation in circumstances including objects with various properties is not implemented. Determining comprises any action or operation by or for a Similarity Comparison **125**, Decision-making Unit **540**, and/or other disclosed elements.

At step **9140**, the first one or more instruction sets for operating the avatar correlated with the first collection of object representations are executed. An instruction set can be executed by a processor and/or other processing element. Executing can be performed in response to the aforementioned determining. In some aspects, an instruction set anticipated or determined to be used for autonomous operating of an avatar of an application may be executed. In other aspects, where an application does not comprise an avatar or rely on an avatar for its operation, an instruction set anticipated or determined to be used for autonomous operating of an application may be executed. Therefore, a reference to an instruction set to be used or executed for autonomous operating of an avatar includes or can be substituted with a reference to an instruction set to be used or executed for autonomous operating of an application depending on context. In some aspects, instruction sets anticipated or determined to be used or executed in an avatar's and/or application's autonomous operation may be referred to as anticipatory instruction sets, alternate instruction sets, and/or other suitable name or reference. Therefore, these terms can be used interchangeably herein depending on context. Executing may include executing one or more alternate instruction sets (i.e. anticipatory instruction sets, etc.) instead of or prior to an instruction set that would have been executed in a regular course of execution. In some embodiments, executing may include modifying a register or other element of a processor with one or more alternate instruction sets. Executing may also include redirecting a processor to one or more alternate instruction sets. In further embodiments, a processor may run an application including instruction sets for operating an avatar and/or the application. In some aspects, executing includes executing one or more alternate instruction sets as part of the avatar and/or application. In other aspects, executing includes modifying the avatar and/or application with one or more alternate instruction sets. In further aspects, executing includes redirecting the avatar and/or application to one or more alternate instruction sets. In further aspects, executing includes modifying one or more instruction sets of the avatar and/or application. In further aspects, executing includes modifying the avatar's and/or application's source code, bytecode, intermediate code, compiled code, interpreted code, translated code, runtime code, assembly code, machine code, or other code. In further aspects, executing includes modifying memory, processor register, storage, repository, and/or other elements where the avatar's and/or application's instruction sets are stored or used. In further aspects, executing includes modifying the avatar and/or application at source code write time, compile time, interpretation time, translation time, linking time, loading time, runtime, or other time. In further aspects, executing includes modifying one or more of the avatar's and/or application's lines of code, statements, instructions, functions, routines, subroutines, basic blocks, or other code segments. In further aspects, executing includes a manual, automatic, dynamic, just in time (JIT), or other instrumentation of the avatar and/or application. In further aspects, executing includes utilizing a dynamic, interpreted, scripting, or other programming language. In further aspects, executing includes utilizing dynamic code, dynamic class loading, or reflection. In further aspects, executing includes utilizing assembly language. In further aspects, executing includes utilizing metaprogramming, self-modifying code, and/or a tool or technique for modify-

ing the avatar and/or application. In further aspects, executing includes utilizing just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further aspects, executing includes utilizing dynamic expression creation, dynamic expression execution, dynamic function creation, or dynamic function execution. In further aspects, executing includes adding or inserting additional code into the avatar's and/or application's code. In further aspects, executing includes modifying, removing, rewriting, or overwriting the avatar's and/or application's code. In further aspects, executing includes branching, redirecting, extending, or hot swapping the avatar's and/or application's code. Branching or redirecting the avatar's and/or application's code may include inserting a branch, jump, or other means for redirecting the avatar's and/or application's execution. In further embodiments, executing includes modifying a virtual machine, a runtime engine, a compiler/interpreter/translator, an operating system, an execution stack, a storage, a memory, an input, and/or other elements of a computing system used in operating an avatar and/or application. Executing may be caused by ACAAO Unit **100**, Artificial Intelligence Unit **110**, Modification Interface **130**, and/or other disclosed elements. Executing comprises any action or operation by or for a Processor **11**, Application Program **18**, Avatar **605**, Modification Interface **130**, and/or other disclosed elements.

At step **9145**, one or more operations defined by the first one or more instruction sets for operating the avatar correlated with the first collection of object representations are performed by the avatar. The one or more operations may be performed in response to the aforementioned executing. In some aspects, an operation includes any operation that can be performed by/with/on an avatar. For example, an operation includes moving, maneuvering, jumping, running, shooting, selecting, and/or other operations in a context of a computer game, virtual world, 3D or 2D graphics application, and/or others. An operation of an avatar may include other operations in contexts of other applications. In further aspects, where an application does not comprise an avatar or rely on an avatar for its operation, an operation includes any operation that can be performed by/with/on an application and/or objects thereof. One of ordinary skill in art will recognize that, while all possible variations of operations by/with/on an avatar and/or application are too voluminous to list and limited only by the avatar's and/or application's design and/or user's utilization, other operations are within the scope of this disclosure.

Referring to FIG. **31**, the illustration shows an embodiment of a method **9200** for learning and/or using an avatar's circumstances for autonomous avatar operation. In some aspects, the method can be used on a computing device or system to enable learning of an avatar's operation in circumstances including objects with various properties and enable autonomous avatar operation in similar circumstances. Method **9200** may include any action or operation of any of the disclosed methods such as method **9100**, **9300**, **9400**, **9500**, **9600**, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method **9200**.

At step **9205**, a first collection of object representations is received. Step **9205** may include any action or operation described in Step **9105** of method **9100** as applicable.

At step **9210**, a first one or more instruction sets for operating an avatar are received. Step **9210** may include any action or operation described in Step **9110** of method **9100** as applicable.

At step **9215**, the first collection of object representations correlated with the first one or more instruction sets for operating the avatar are learned. Step **9215** may include any action or operation described in Step **9115** and/or Step **9120** of method **9100** as applicable.

At step **9220**, a new collection of object representations is received. Step **9220** may include any action or operation described in Step **9125** of method **9100** as applicable.

At step **9225**, the first one or more instruction sets for operating the avatar correlated with the first collection of object representations are anticipated based on at least a partial match between the new collection of object representations and the first collection of object representations. Step **9225** may include any action or operation described in Step **9130** and/or Step **9135** of method **9100** as applicable.

At step **9230**, the first one or more instruction sets for operating the avatar correlated with the first collection of object representations are executed. Step **9230** may include any action or operation described in Step **9140** of method **9100** as applicable.

At step **9235**, one or more operations defined by the first one or more instruction sets for operating the avatar correlated with the first collection of object representations are performed by the avatar. Step **9235** may include any action or operation described in Step **9145** of method **9100** as applicable.

Referring to FIG. **32**, the illustration shows an embodiment of a method **9300** for learning and/or using an avatar's circumstances for autonomous avatar operation. In some aspects, the method can be used on a computing device or system to enable learning of an avatar's operation in circumstances including objects with various properties and enable autonomous avatar operation in similar circumstances. Method **9300** may include any action or operation of any of the disclosed methods such as method **9100**, **9200**, **9400**, **9500**, **9600**, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method **9300**.

At step **9305**, a first stream of collections of object representations is received. Step **9305** may include any action or operation described in Step **9105** of method **9100** as applicable.

At step **9310**, a first one or more instruction sets for operating an avatar are received. Step **9310** may include any action or operation described in Step **9110** of method **9100** as applicable.

At step **9315**, the first stream of collections of object representations is correlated with the first one or more instruction sets for operating the avatar. Step **9315** may include any action or operation described in Step **9115** of method **9100** as applicable.

At step **9320**, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar are stored. Step **9320** may include any action or operation described in Step **9120** of method **9100** as applicable.

At step **9325**, a new stream of collections of object representations is received. Step **9325** may include any action or operation described in Step **9125** of method **9100** as applicable.

At step **9330**, the new stream of collections of object representations is compared with the first stream of collections of object representations. Step **9330** may include any action or operation described in Step **9130** of method **9100** as applicable.

At step **9335**, a determination is made that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. Step **9335** may include any action or operation described in Step **9135** of method **9100** as applicable.

At step **9340**, the first one or more instruction sets for operating the avatar correlated with the first stream of collections of object representations are executed. Step **9340** may include any action or operation described in Step **9140** of method **9100** as applicable.

At step **9345**, one or more operations defined by the first one or more instruction sets for operating the avatar correlated with the first stream of collections of object representations are performed by the avatar. Step **9345** may include any action or operation described in Step **9145** of method **9100** as applicable.

Referring to FIG. **33**, the illustration shows an embodiment of a method **9400** for learning and/or using an application's circumstances for autonomous application operation. In some aspects, the method can be used on a computing device or system to enable learning of an application's operation in circumstances including objects with various properties and enable autonomous application operation in similar circumstances. Method **9400** may include any action or operation of any of the disclosed methods such as method **9100**, **9200**, **9300**, **9500**, **9600**, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method **9400**.

At step **9405**, a first collection of object representations is received. Step **9405** may include any action or operation described in Step **9105** of method **9100** as applicable.

At step **9410**, a first one or more instruction sets for operating an application are received. Step **9410** may include any action or operation described in Step **9110** of method **9100** as applicable.

At step **9415**, the first collection of object representations is correlated with the first one or more instruction sets for operating the application. Step **9415** may include any action or operation described in Step **9115** of method **9100** as applicable.

At step **9420**, the first collection of object representations correlated with the first one or more instruction sets for operating the application are stored. Step **9420** may include any action or operation described in Step **9120** of method **9100** as applicable.

At step **9425**, a new collection of object representations is received. Step **9425** may include any action or operation described in Step **9125** of method **9100** as applicable.

At step **9430**, the new collection of object representations is compared with the first collection of object representations. Step **9430** may include any action or operation described in Step **9130** of method **9100** as applicable.

At step **9435**, a determination is made that there is at least a partial match between the new collection of object representations and the first collection of object representations. Step **9435** may include any action or operation described in Step **9135** of method **9100** as applicable.

At step **9440**, the first one or more instruction sets for operating the application correlated with the first collection of object representations are executed. Step **9440** may include any action or operation described in Step **9140** of method **9100** as applicable.

At step **9445**, one or more operations defined by the first one or more instruction sets for operating the application correlated with the first collection of object representations are performed by the application. Step **9445** may include any action or operation described in Step **9145** of method **9100** as applicable.

Referring to FIG. **34**, the illustration shows an embodiment of a method **9500** for learning and/or using an application's circumstances for autonomous application operation. In some aspects, the method can be used on a computing device or system to enable learning of an application's operation in circumstances including objects with various properties and enable autonomous application operation in similar circumstances. Method **9500** may include any action or operation of any of the disclosed methods such as method **9100**, **9200**, **9300**, **9400**, **9600**, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method **9500**.

At step **9505**, a first collection of object representations is received. Step **9505** may include any action or operation described in Step **9105** of method **9100** as applicable.

At step **9510**, a first one or more instruction sets for operating an application are received. Step **9510** may include any action or operation described in Step **9110** of method **9100** as applicable.

At step **9515**, the first collection of object representations correlated with the first one or more instruction sets for operating the application are learned. Step **9515** may include any action or operation described in Step **9115** and/or Step **9120** of method **9100** as applicable.

At step **9520**, a new collection of object representations is received. Step **9520** may include any action or operation described in Step **9125** of method **9100** as applicable.

At step **9525**, the first one or more instruction sets for operating the application correlated with the first collection of object representations are anticipated based on at least a partial match between the new collection of object representations and the first collection of object representations. Step **9525** may include any action or operation described in Step **9130** and/or Step **9135** of method **9100** as applicable.

At step **9530**, the first one or more instruction sets for operating the application correlated with the first collection of object representations are executed. Step **9530** may include any action or operation described in Step **9140** of method **9100** as applicable.

At step **9535**, one or more operations defined by the first one or more instruction sets for operating the application correlated with the first collection of object representations are performed by the application. Step **9535** may include any action or operation described in Step **9145** of method **9100** as applicable.

Referring to FIG. **35**, the illustration shows an embodiment of a method **9600** for learning and/or using an application's circumstances for autonomous application operation. In some aspects, the method can be used on a computing device or system to enable learning of an application's operation in circumstances including objects with various properties and enable autonomous application operation in similar circumstances. Method **9600** may

include any action or operation of any of the disclosed methods such as method **9100**, **9200**, **9300**, **9400**, **9500**, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method **9600**.

At step **9605**, a first stream of collections of object representations is received. Step **9605** may include any action or operation described in Step **9105** of method **9100** as applicable.

At step **9610**, a first one or more instruction sets for operating an application are received. Step **9610** may include any action or operation described in Step **9110** of method **9100** as applicable.

At step **9615**, the first stream of collections of object representations is correlated with the first one or more instruction sets for operating the application. Step **9615** may include any action or operation described in Step **9115** of method **9100** as applicable.

At step **9620**, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the application are stored. Step **9620** may include any action or operation described in Step **9120** of method **9100** as applicable.

At step **9625**, a new stream of collections of object representations is received. Step **9625** may include any action or operation described in Step **9125** of method **9100** as applicable.

At step **9630**, the new stream of collections of object representations is compared with the first stream of collections of object representations. Step **9630** may include any action or operation described in Step **9130** of method **9100** as applicable.

At step **9635**, a determination is made that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. Step **9635** may include any action or operation described in Step **9135** of method **9100** as applicable.

At step **9640**, the first one or more instruction sets for operating the application correlated with the first stream of collections of object representations are executed. Step **9640** may include any action or operation described in Step **9140** of method **9100** as applicable.

At step **9645**, one or more operations defined by the first one or more instruction sets for operating the application correlated with the first stream of collections of object representations are performed by the application. Step **9645** may include any action or operation described in Step **9145** of method **9100** as applicable.

Referring to FIG. **36**, in some exemplary embodiments, Application Program **18** may be or include a 3D Computer Game **18***a*. Examples of 3D Computer Game **18***a* include a first shooter game, a flight simulation, a driving simulation, and/or others. Avatar **605** may be or include Soldier **605***a* within 3D Computer Game **18***a*. Soldier **605***a* can be controlled by User **50** (i.e. game player, etc.) through inputting operating directions via Human-machine Interface **23** such as a game controller, keyboard, joystick, or other input device. For instance, responsive to User's **50** manipulating one or more game controller elements, Soldier **605***a* may be caused to move, maneuver, shoot, jump, and/or perform other operations. Computing Device **70** may include or be coupled to ACAAO Unit **100**. ACAAO Unit **100** can obtain objects (i.e. Opponent **615***aa*, Dog **615***ab*, Pedestrian **615***ac*, etc.) and/or their properties in Soldier's **605***a* surrounding within 3D Computer Game **18***a*. ACAAO Unit **100** may create or generate one or more (i.e. stream, etc.) Collections of Object Representations **525** comprising Object Representations **625**, Object Properties **630**, and/or other elements or information representing Objects **615** in Soldier's **605***a* surrounding. ACAAO Unit **100** can also obtain Instruction Sets **526** used or executed in operating Soldier **605***a*. ACAAO Unit **100** can also optionally obtain any Extra Info **527** (i.e. time, location, computed, visual, acoustic, contextual, and/or other information, etc.) related to Soldier's **605***a* operation. As User **50** operates Soldier **605***a* in circumstances including objects with various properties as shown, ACAAO Unit **100** may learn Soldier's **605***a* operation in these circumstances by correlating Collections of Object Representations **525** representing Objects **615** in Soldier's **605***a* surrounding with one or more Instruction Sets **526** used or executed in operating Soldier **605***a*. Any Extra Info **527** related to Soldier's **605***a* operation may also optionally be correlated with Collections of Object Representations **525**. ACAAO Unit **100** can store this knowledge into Knowledgebase **530** (i.e. Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.). In the future, ACAAO Unit **110** may compare incoming Collections of Object Representations **525** representing Objects **615** in Soldier's **605***a* surrounding with previously learned Collections of Object Representations **525** including optionally using any Extra Info **527** for enhanced decision making. If substantially similar or at least a partial match is found or determined, ACAAO Unit **110** may cause the Instruction Sets **526** correlated with the previously learned Collections of Object Representations **525** to be executed, thereby enabling autonomous operation of Soldier **605***a* in similar circumstances as in previously learned ones. For instance, ACAAO Unit **100** may learn User **50**-directed shooting at Opponent **615***aa* by Soldier **605***a* in a circumstance that includes Opponent **615***aa*, Dog **615***ab*, Pedestrian **615***ac*, and/or other Objects **615** among which Soldier **605***a* may need to maneuver and/or with which Soldier **605***a* may need to interact, as shown. In the future, when a circumstance that includes one or more Objects **615** with similar Object Properties **630** is encountered, ACAAO Unit **100** may implement the shooting at Opponent **615***aa* by Soldier **605***a* autonomously. In some designs, 3D Computer Game **18***a* may include elevated Objects **615** such as flying objects (i.e. flying animals, aircraft, etc.), objects on hills or mountains, objects on buildings, and/or others in which case altitudinal information related to distance and bearing/angle of Objects **615** relative to Soldier **605***a* can be obtained, learned, and used. In other designs, the street (not enumerated), parts thereof (i.e. curbs, sidewalks, etc.), and/or objects thereon (i.e. buildings, etc.) can be obtained as Objects **615** themselves, which may be learned and used.

In some embodiments, ACAAO Unit **100** may reside on Server **96** accessible over Network **95** as previously described. In such embodiments, any number of Computing Devices **70**, Processors **11**, 3D Computer Games **18***a*, and/or other elements may connect to such remote ACAAO Unit **100** and the remote ACAAO Unit **100** may learn operations of their Soldiers **605***a* in circumstances including objects with various properties. In turn, any number of Computing Devices **70**, Processors **11**, 3D Computer Games **18***a*, and/or other elements can utilize the remote ACAAO Unit **100** for autonomous operation of their Soldiers **605***a*. For example, multiple Users **50** (i.e. game players, etc.) may operate their Soldiers **605***a* in 3D Computer Game **1** a running on their respective Computing Devices **70** where the Computing

Devices **70** and/or elements thereof may be configured to transmit Soldiers' **605***a* operations in circumstances including objects with various properties to a remote ACAAO Unit **100**. Such remote ACAAO Unit **100** enables learning of the game players' collective knowledge of operating Soldier **605***a* in circumstances including objects with various properties. Any of the disclosed elements such as Artificial Intelligence Unit **110**, Knowledgebase **530**, and/or others can reside on Server **96**, and any combination of local and remote elements can be implemented in alternate embodiments.

In some embodiments, a combination of ACAAO Unit **100** and other systems and/or techniques can be utilized to implement Soldier's **605***a* operation. In one example, ACAAO Unit **100** can be a primary or preferred system for implementing Soldier's **605***a* operation. While operating autonomously under the control of ACAAO Unit **100**, Soldier **605***a* may encounter a circumstance including objects with various properties that has not been encountered or learned before. In such situations, User **50** and/or non-ACAAO system may take control of Soldier's **605***a* operation. ACAAO Unit **100** may take control again when Soldier **605***a* encounters a previously learned circumstance including objects with various properties. Naturally, ACAAO Unit **100** can learn Soldier's **605***a* operation in circumstances while User **50** and/or non-ACAAO system is in control of Soldier **605***a*, thereby reducing or eliminating the need for future involvement of User **50** and/or non-ACAAO system. In another example, User **50** and/or non-ACAAO system can be a primary or preferred system for implementing Soldier's **605***a* operation. While operating under the control of User **50** and/or non-ACAAO system, User **50** and/or non-ACAAO system may release control to ACAAO Unit **100** for any reason (i.e. User **50** gets tired or distracted, non-ACAAO system gets stuck or cannot make a decision, etc.), at which point Soldier **605***a* can be controlled by ACAAO Unit **100**. In some designs, ACAAO Unit **100** may take control in certain special circumstances including objects with various properties where ACAAO Unit **100** offers superior performance even though User **50** and/or non-ACAAO system may generally be preferred. Once Soldier **605***a* leaves such special circumstances, ACAAO Unit **100** may release control to User **50** and/or non-ACAAO system. In general, ACAAO Unit **100** can take control from, share control with, or release control to User **50**, non-ACAAO system, and/or other system or process at any time, in any circumstances, and remain in control for any period of time as needed.

In some embodiments, ACAAO Unit **100** may control one or more elements of Soldier **605***a* while User **50** and/or non-ACAAO system may control other one or more elements of Soldier **605***a*. For example, ACAAO Unit **100** may control Soldier's **605***a* movement, while User **50** and/or non-ACAAO system may control Soldier's **605***a* aiming and shooting. Any other combination of controlling various elements or functions of Soldier **605***a* by ACAAO Unit **100**, User **50**, and/or non-ACAAO system can be implemented.

In some embodiments, ACAAO Unit **100** enables learning of a particular User's **50** (i.e. game player's, etc.) knowledge, methodology, or style of operating Soldier **605***a* within 3D Game Application **18***a*. In some aspects, learning of a particular User's **50** knowledge, methodology, or style of operating Soldier **605***a* includes learning the User's **50** directing or operating Soldier **605***a* in circumstances including objects with various properties. In one example, one User **50** may shoot an opponent while another User **50** may strike the opponent with a sword. In another example, one User **50** may jump over an obstacle while another User **50** may move around the obstacle, and so on. The knowledge of User's **50** methodology or style of operating Soldier **605***a* can be used to enable personalized autonomous operation of Soldier **605***a* specific to a particular User **50**. Therefore, ACAAO-enabled Soldier **605***a* can exemplify User's **50** knowledge, methodology, or style of operating Soldier **605***a* as learned from User **50**. In some aspects, this functionality enables one or more ACAAO-enabled Soldiers **605***a* to be utilized in 3D Computer Game **18***a* to assist User **50** in defeating an opponent or achieving other goals. In one example, User **50** can utilize a team of ACAAO-enabled Soldiers **605***a* each of which may exemplify User's **50** knowledge, methodology, or style of operating Soldier **605***a*. In one instance, ACAAO-enabled Soldiers **605***a* may be dispersed around a User **50**-controlled Soldier **605***a* within a specific radius and follow User **50**-controlled Soldier's **605***a* movement. In another instance, ACAAO-enabled Soldiers **605***a* may move autonomously toward a certain point or goal in 3D Computer Game **18***a*. In a further instance, ACAAO-enabled Soldiers **605***a* can be completely autonomous and rely solely on the knowledge learned from User's **50** methodology or style of operating Soldier **605***a*. In other aspects, ACAAO Unit **100** enables a professional or other experienced User **50** (i.e. game player, etc.) to record his/her knowledge, methodology, or style of operating Soldier **605***a* into Knowledgebase **530** (i.e. Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.) and/or other repository. User **50** can then sell or make available his/her knowledge, methodology, or style of operating Soldier **605***a* to other users who may want to implement User's **50** knowledge, methodology, or style of operating Soldier **605***a*. Knowledgebase **530** and/or other repository comprising User's **50** knowledge, methodology, or style of operating Soldier **605***a* can be available to other users via a storage medium, via a network, or via other means.

One of ordinary skill in art will understand that the aforementioned features, functionalities, and embodiments described with respect to 3D Computer Game **18***a* can be implemented in any 3D Application Program **18** such as a 3D virtual world, 3D graphics application, computer aided design (CAD) application, and/or others. Similar features, functionalities, and embodiments can also be implemented in a 2D Application Program **18**, and/other application program as applicable, and vice versa.

Referring to FIG. **37**, in some exemplary embodiments, Application Program **18** may be or include a 2D Computer Game **18***b*. Examples of 2D Computer Game **18***b* include a strategy game, a shooter game, a tile-matching game, a platform game, and/or others. Avatar **605** may be or include Tank **605***b* within 2D Computer Game **18***b*. Tank **605***b* can be controlled by User **50** (i.e. game player, etc.) through inputting operating directions via Human-machine Interface **23** such as a game controller, keyboard, joystick, or other input device. For instance, responsive to User's **50** manipulating one or more game controller elements, Tank **605***b* may be caused to move, maneuver, shoot, and/or perform other operations. Computing Device **70** may include or be coupled to ACAAO Unit **100**. ACAAO Unit **100** can obtain objects (i.e. Helicopter **615***ba*, Rocket Launcher **615***bb*, Communication Center **615***bc*, Tank **615***bd*, etc.) and/or their properties in Tank's **605***b* surrounding within 2D Computer Game **18***b*. ACAAO Unit **100** may create or generate one or more (i.e. stream, etc.) Collections of Object Representations **525** comprising Object Representations **625**, Object Properties **630**, and/or other elements or information representing

Objects **615** in Tank's **605***b* surrounding. ACAAO Unit **100** can also obtain Instruction Sets **526** used or executed in operating Tank **605***b*. ACAAO Unit **100** can also optionally obtain any Extra Info **527** (i.e. time, location, computed, visual, acoustic, contextual, and/or other information, etc.) related to Tank's **605***b* operation. As User **50** operates Tank **605***b* in circumstances including objects with various properties as shown, ACAAO Unit **100** may learn Tank's **605***b* operation in these circumstances by correlating Collections of Object Representations **525** representing Objects **615** in Tank's **605***b* surrounding with one or more Instruction Sets **526** used or executed in operating Tank **605***b*. Any Extra Info **527** related to Tank's **605***b* operation may also optionally be correlated with Collections of Object Representations **525**. ACAAO Unit **100** can store this knowledge into Knowledgebase **530** (i.e. Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.). In the future, ACAAO Unit **110** may compare incoming Collections of Object Representations **525** representing Objects **615** in Tank's **605***b* surrounding with previously learned Collections of Object Representations **525** including optionally using any Extra Info **527** for enhanced decision making. If substantially similar or at least a partial match is found or determined, ACAAO Unit **110** may cause the Instruction Sets **526** correlated with the previously learned Collections of Object Representations **525** to be executed, thereby enabling autonomous operation of Tank **605***b* in similar circumstances as in previously learned ones. For instance, ACAAO Unit **100** may learn User **50**-directed maneuvering and shooting by Tank **605***b* in a circumstance that includes Helicopter **615***ba*, Rocket Launcher **615***bb*, Communication Center **615***bc*, Tank **615***bd*, and/or other Objects **615** among which Tank **605***b* may need to maneuver and/or with which Tank **605***b* may need to interact, as shown. In the future, when a circumstance that includes one or more Objects **615** with similar Object Properties **630** is encountered, ACAAO Unit **100** may implement the maneuvering and shooting by Tank **605***b* autonomously. In some designs, 2D Computer Game **18***b* may include 3D effects such as the effect of flying objects (i.e. flying animals, aircraft, Helicopter **615***ba*, etc.), objects on hills or mountains, objects on buildings, and/or other elevated objects in which case altitudinal information related to distance and bearing/angle of Objects **615** relative to Tank **605***b* can be obtained, learned, and used. In other designs, the canyon (not enumerated) and/or parts thereof (i.e. cliffs, ridges, etc.) can be obtained as Objects **615** themselves, which may be learned and used.

Referring to FIG. **38**, in some exemplary embodiments, an Area of Interest **450** can be utilized. In one example, Area of Interest **450** may include a radial, circular, elliptical, or other such area around Tank **605***b*. In another example, Area of Interest **450** may include a triangular, rectangular, octagonal, or other such area around Tank **605***b*. In a further example, Area of Interest **450** may include a spherical, cubical, pyramid-like, or other such area around Tank **605***b* as applicable to 3D space. Any other Area of Interest **450** shape can be utilized depending on implementation. The shape and/or size of Area of Interest **450** can be defined by a user, by ACAAO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. For instance, Area of Interest **450** may include or be defined by a circle around Tank **605***b* with a radius of 250 meters. Any other radiuses or sizes of Area of Interest **450** can be used such as 0.27 m, 1 m, 7 m, 19 m, 382 m, 7116 m, 49276 m, and so on. Utilizing Area of Interest **450** enables ACAAO Unit **100** to focus on Tank's **605***b* surrounding, thereby ignoring extraneous detail in the rest of the space. In some designs, Tank's **605***b* surrounding may include or be defined by Area of Interest **450**. In some aspects, Area of Interest **450** can be subdivided into sub-areas (i.e. sub-circles, sub-rectangles, sub-spheres, etc.). Sub-areas can be used to classify the surrounding by distance from Tank **605***b*. For example, the surrounding closer to Tank **605***b* may be more important and may be assigned higher importance index or weight. As User **50** operates Tank **605***b* in circumstances including objects with various properties as shown, ACAAO Unit **100** may learn Tank's **605***b* operation in these circumstances by correlating Collections of Object Representations **525** representing Objects **615** in Area of Interest **450** around Tank **605***b* with one or more Instruction Sets **526** used or executed in operating Tank **605***b*. Any Extra Info **527** related to Tank's **605***b* operation may also optionally be correlated with Collections of Object Representations **525**. ACAAO Unit **100** can store this knowledge into Knowledgebase **530** (i.e. Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.). In the future, ACAAO Unit **110** may compare incoming Collections of Object Representations **525** representing Objects **615** in Area of Interest **450** around Tank **605***b* with previously learned Collections of Object Representations **525** including optionally using any Extra Info **527** for enhanced decision making. If substantially similar or at least a partial match is found or determined, ACAAO Unit **110** may cause the Instruction Sets **526** correlated with the previously learned Collections of Object Representations **525** to be executed, thereby enabling autonomous operation of Tank **605***b* in similar Areas of Interest **450** as in previously learned ones. For instance, ACAAO Unit **100** may learn User **50**-directed maneuvering and shooting by Tank **605***b* in an Area of Interest **450** that includes Rocket Launcher **615***bb*, Communication Center **615***bc*, Tank **615***bd*, and/or other Objects **615** among which Tank **605***b* may need to maneuver and/or with which Tank **605***b* may need to interact, as shown. In the future, when an Area of Interest **450** that includes one or more Objects **615** with similar Object Properties **630** is encountered, ACAAO Unit **100** may implement the maneuvering and shooting by Tank **605***b* autonomously.

Referring to FIG. **39**, in some exemplary embodiments, Application Program **18** may be or include a 2D Computer Game **18***c* comprising multiple Avatars **605** that User **50** can control or operate. As multiple Avatars **605** can perform operations in such 2D Computer Game **18***c*, Area of Interest **450** in 2D Computer Game **18***c* including multiple Avatars **605** and/or other Objects **615** may be more relevant than a particular Avatar's **605** surrounding or Area of Interest **450** for learning circumstances including objects with various properties. In some aspects, Area of Interest **450** in 2D Computer Game **18***c* may include the entire 2D Computer Game **18***c*, a part of 2D Computer Game **18***c* that is shown to User **50** (i.e. on a display, via a graphical user interface, etc.), or any part of 2D Computer Game **18***c*. Examples of 2D Computer Game **18***c* include a strategy game, a board game, a tile-matching game, and/or others. Avatars **605** may be or include Ships **605***c***1**-**605***c***3**, etc. within 2D Computer Game **18***c*. Ships **605***c***1**-**605***c***3**, etc. can be controlled by User **50** (i.e. game player, etc.) through inputting operating directions via Human-machine Interface **23** such as a game controller, keyboard, joystick, or other input device. For instance, responsive to User's **50** manipulating one or more game controller elements, Ships **605***c***1**-**605***c***3**, etc. may be caused to move, maneuver, shoot, and/or perform other

operations. Computing Device **70** may include or be coupled to ACAAO Unit **100**. ACAAO Unit **100** can obtain objects and/or their properties in Area of Interest **450** in 2D Computer Game **18***c*. For instance, Area of Interest **450** in 2D Computer Game **18***c* may include Ships **605***c***1**-**605***c***3**, etc., Ships **615***c***1**-**615***c***3**, etc., and/or other objects. Properties of each of the objects may include a location defined by coordinates. ACAAO Unit **100** may create or generate one or more (i.e. stream, etc.) Collections of Object Representations **525** comprising Object Representations **625**, Object Properties **630**, and/or other elements or information representing Objects **615** in Area of Interest **450** in 2D Computer Game **18***c*. ACAAO Unit **100** can also obtain Instruction Sets **526** used or executed in operating Ships **605***c***1**-**605***c***3**, etc. ACAAO Unit **100** can also optionally obtain any Extra Info **527** (i.e. time, location, computed, visual, acoustic, contextual, and/or other information, etc.) related to Ships' **605***c***1**-**605***c***3**, etc. operation. As User **50** operates Ships **605***c***1**-**605***c***3**, etc. in circumstances including objects with various properties as shown, ACAAO Unit **100** may learn Ships' **605***c***1**-**605***c***3**, etc. operation in these circumstances by correlating Collections of Object Representations **525** representing Objects **615** in Area of Interest **450** in 2D Computer Game **18***c* with one or more Instruction Sets **526** used or executed in operating Ships **605***c***1**-**605***c***3**, etc. Any Extra Info **527** related to Ships' **605***c***1**-**605***c***3**, etc. operation may also optionally be correlated with Collections of Object Representations **525**. ACAAO Unit **100** can store this knowledge into Knowledgebase **530** (i.e. Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.). In the future, ACAAO Unit **110** may compare incoming Collections of Object Representations **525** representing Objects **615** in Area of Interest **450** in 2D Computer Game **18***c* with previously learned Collections of Object Representations **525** including optionally using any Extra Info **527** for enhanced decision making. If substantially similar or at least a partial match is found or determined, ACAAO Unit **110** may cause the Instruction Sets **526** correlated with the previously learned Collections of Object Representations **525** to be executed, thereby enabling autonomous operation of Ships **605***c***1**-**605***c***3**, etc. in similar circumstances as in previously learned ones. For instance, ACAAO Unit **100** may learn User **50**-directed maneuvering and shooting by Ships **605***c***1**-**605***c***3**, etc. in a circumstance that includes Ships **615***c***1**-**615***c***3** and/or other Objects **615** among which Ships **605***c***1**-**605***c***3**, etc. may need to maneuver and/or with which Ships **605***c***1**-**605***c***3**, etc. may need to interact, as shown. In the future, when a circumstance that includes one or more Objects **615** with similar Object Properties **630** is encountered, ACAAO Unit **100** may implement the maneuvering and shooting by Ships **605***c***1**-**605***c***3**, etc. autonomously. In some designs, the islands (not enumerated), parts thereof (i.e. cliffs, ridges, beaches, mountains, etc.), and/or objects thereon (i.e. buildings, etc.) can be obtained as Objects **615** themselves, which may be learned and used.

One of ordinary skill in art will understand that the features, functionalities, and embodiments described in the above exemplary embodiments with respect to Soldier **605***a*, Tank **605***b*, and Ships **605***c***1**-**605***c***3** can similarly be implemented by/with/on any Avatar **605** of Application Program **18** or by/with/on any Application Program **18**.

In some exemplary embodiments (not depicted), Application Program **18** may not comprise an Avatar **605** and/or may not rely on Avatar **605** for its operation. Application Program **18** can be controlled by User **50** through inputting operating directions via Human-machine Interface **23** such as a mouse, keyboard, or other input device. For instance, responsive to User's **50** pressing one or more mouse buttons, moving the mouse, and/or pressing keyboard buttons, an Object **615** of Application Program **18** may be inserted, deleted, selected, moved, and/or subject to other operations. Computing Device **70** may include or be coupled to ACAAO Unit **100**. ACAAO Unit **100** can obtain objects and/or their properties in Application Program **18**. In some aspects, Area of Interest **450** in Application Program **18** can be utilized as previously described. Area of Interest **450** in Application Program **18** may include the entire Application Program **18**, a part of Application Program **18** that is shown to User **50** (i.e. on a display, via a graphical user interface, etc.), or any part of Application Program **18**. For instance, ACAAO Unit **100** can obtain Objects **615** in Application Program **18** shown to User **50** via a graphical user interface. ACAAO Unit **100** may create or generate one or more (i.e. stream, etc.) Collections of Object Representations **525** comprising Object Representations **625**, Object Properties **630**, and/or other elements or information representing Objects **615** in Application Program **18**. ACAAO Unit **100** can also obtain Instruction Sets **526** used or executed in operating Application Program **18** and/or Objects **615** thereof. ACAAO Unit **100** can also optionally obtain any Extra Info **527** (i.e. time, location, computed, visual, acoustic, contextual, and/or other information, etc.) related to Application Program's **18** and/or Objects' **615** thereof operation. As User **50** operates Application Program **18** in circumstances including objects with various properties, ACAAO Unit **100** may learn Application Program's **18** operation in these circumstances by correlating Collections of Object Representations **525** representing Objects **615** in Application Program **18** with one or more Instruction Sets **526** used or executed in operating Application Program **18** and/or Objects **615** thereof. Any Extra Info **527** related to Application Program's **18** and/or Objects' **615** thereof operation may also optionally be correlated with Collections of Object Representations **525**. ACAAO Unit **100** can store this knowledge into Knowledgebase **530** (i.e. Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.). In the future, ACAAO Unit **110** may compare incoming Collections of Object Representations **525** representing Objects **615** in Application Program **18** with previously learned Collections of Object Representations **525** including optionally using any Extra Info **527** for enhanced decision making. If substantially similar or at least a partial match is found or determined, ACAAO Unit **110** may cause the Instruction Sets **526** correlated with the previously learned Collections of Object Representations **525** to be executed, thereby enabling autonomous operation of Application Program **18** and/or Objects **615** thereof in similar circumstances as in previously learned ones. For instance, ACAAO Unit **100** may learn User **50**-directed selecting an Object **615**, moving an Object **615**, and/or deleting an Object **615** of Application Program **18** in a circumstance that includes various Objects **615**. In the future, when a circumstance that includes one or more Objects **615** with similar Object Properties **630** is encountered, ACAAO Unit **100** may implement the selecting an Object **615**, moving an Object **615**, and/or deleting an Object **615** of Application Program **18** autonomously.

One of ordinary skill in art will understand that the functionalities described with respect to Application Program **18** and/or Objects **615** thereof can be implemented in any Application Program **18** such as a computer game, a virtual world, a 2D or 3D graphics application, a web browser, a media application, a word processing application,

a spreadsheet application, a database application, a forms-based application, an operating system, a device/system control application, and/or others as applicable. In such Application Programs **18**, examples of Objects **615** that can be utilized include a 2D model, a 3D model, a 2D shape (i.e. point, line, square, rectangle, circle, triangle, etc.), a 3D shape (i.e. cube, sphere, etc.), a graphical user interface (GUI) element, a form element (i.e. text field, radio button, push button, check box, etc.), a data or database element, a spreadsheet element, a link, a picture, a text (i.e. character, word, etc.), a number, and/or others as applicable. Application Program **18** may be or include any Application Program **18** that can benefit from the functionalities described herein.

It must be noted that as used herein and in the appended claims, the singular forms "a", "an", and "the" include plural referents unless the context dearly dictates otherwise.

A number of embodiments have been described herein. While this disclosure contains many specific implementation details, these should not be construed as limitations on the scope of the disclosure or of what may be claimed, but rather as descriptions of features specific to particular embodiments. It should be understood that various modifications can be made without departing from the spirit and scope of the disclosure. The logic flows depicted in the figures do not require the particular order shown, or sequential order, to achieve desirable results. Other or additional steps, elements, or connections can be included, or some of the steps, elements, or connections can be excluded, or a combination thereof can be utilized in the described flows, illustrations, or descriptions. Further, the various aspects of the disclosed devices, apparatuses, systems, and/or methods can be combined in whole or in part with each other to produce additional implementations. Moreover, separation of various components in the embodiments described herein should not be understood as requiring such separation in all embodiments, and it should be understood that the described components can generally be integrated together in a single product or packaged into multiple products. Accordingly, other embodiments are within the scope of the following claims.

The invention claimed is:

1. A system comprising:

one or more processors; and

one or more non-transitory machine readable media storing machine readable code that, when executed by the one or more processors, causes the one or more processors to perform at least:

accessing a first correlation including a first one or more object representations correlated with a first one or more instruction sets for operating a first avatar of an application, wherein the first one or more object representations represent one or more objects of the application;

generating or receiving a second one or more object representations, wherein the second one or more object representations represent one or more objects of the application;

determining the first one or more instruction sets for operating the first avatar of the application based on at least partial match between the second one or more object representations and the first one or more object representations; and

at least in response to the determining, causing the first avatar of the application or a second avatar of the application to perform one or more operations defined by the first one or more instruction sets for operating the first avatar of the application at least by executing the first one or more instruction sets for operating the first avatar of the application.

2. The system of claim **1**, wherein the first one or more object representations include: one object representation, a stream of object representations, a collection of object representations, or a stream of collections of object representations, and wherein the second one or more object representations include: one object representation, a stream of object representations, a collection of object representations, or a stream of collections of object representations, and wherein the one or more objects of the application represented by the first one or more object representations are detected at a first time or during a first time period, and wherein the one or more objects of the application represented by the second one or more object representations are detected at a second time or during a second time period.

3. The system of claim **1**, wherein the determining the first one or more instruction sets for operating the first avatar of the application based on the at least partial match between the second one or more object representations and the first one or more object representations includes:

determining that a number of at least partially matching portions of the second one or more object representations and portions of the first one or more object representations exceeds a threshold number, or

determining that a percentage of at least partially matching portions of the second one or more object representations and portions of the first one or more object representations exceeds a threshold percentage.

4. The system of claim **1**, wherein the first correlation is included in a knowledgebase, and wherein the knowledgebase further includes a third correlation including a third one or more object representations correlated with a third one or more instruction sets for operating: the first avatar of the application, the second avatar of the application, a third avatar of the application, or an avatar of another application, and wherein the third one or more object representations represent one or more objects of: the application, or the another application, and wherein the first correlation is connected with the third correlation by a connection.

5. The system of claim **1**, wherein the second avatar of the application performs the one or more operations defined by the first one or more instruction sets for operating the first avatar of the application.

6. The system of claim **1**, wherein the one or more objects of the application represented by the first one or more object representations include one or more objects of the application in the first avatar's surrounding, and wherein the one or more objects of the application represented by the second one or more object representations include: one or more objects of the application in the first avatar's surrounding, or one or more objects of the application in the second avatar's surrounding, and wherein the first avatar's surrounding includes at least one of:

a part of the application in an area of interest around the first avatar,

a part of the application defined by a threshold distance from the first avatar,

a part of the application relative to the first avatar,

a part of the application around the first avatar,

a part of the application that is shown to a user,

a part of the application that is visible to a user, or

a part of the application, and wherein the second avatar's surrounding includes at least one of:

a part of the application in an area of interest around the second avatar,

a part of the application defined by a threshold distance from the second avatar,
a part of the application relative to the second avatar,
a part of the application around the second avatar,
a part of the application that is shown to a user,
a part of the application that is visible to a user, or
a part of the application.

7. The system of claim **1**, wherein, to correlate the first one or more object representations with the first one or more instruction sets for operating the first avatar of the application, a determination is made that the first one or more instruction sets for operating the first avatar of the application temporally correspond to the first one or more object representations.

8. The system of claim **1**, wherein at least some elements of the system are included in: a single device, or multiple devices, and wherein the one or more processors include: one or more microcontrollers, one or more computing devices, or one or more electronic devices, and wherein the first correlation is stored in or on at least one of: at least one non-transitory machine readable medium of the one or more non-transitory machine readable media, another one or more non-transitory machine readable media, one or more volatile memories, one or more non-volatile memories, one or more storage devices, or one or more storage systems, and wherein the application includes: a video game, a computer game, a simulation program, a program including text processing, a program including number processing, a program including picture processing, a program including object processing, or a program, and wherein the application includes: one or more versions of the application, one or more upgrades of the application, one or more sequels of the application, one or more instances of the application, or one or more variations of the application, and wherein the first avatar of the application is a first object of the application and the second avatar of the application is a second object of the application, and wherein the first one or more object representations include one or more properties of the one or more objects represented by the first one or more object representations, and wherein the second one or more object representations include one or more properties of the one or more objects represented by the second one or more object representations, and wherein an instruction set of the first one or more instruction sets for operating the first avatar of the application includes at least one of: only one instruction, a plurality of instructions, one or more inputs, one or more commands, one or more computer commands, one or more keywords, one or more symbols, one or more operators, one or more variables, one or more values, one or more objects, one or more object references, one or more data structures, one or more data structure references, one or more functions, one or more function references, one or more parameters, one or more signals, one or more characters, one or more digits, one or more numbers, one or more user operating directions, one or more user directions, one or more user inputs, one or more representations of one or more user actions, one or more representations of one or more user clicks, one or more binary bits, one or more assembly language commands, one or more states, one or more state representations, one or more codes, one or more data, or one or more information, and wherein: an object of the application of the one or more objects of the application represented by the first one or more object representations is the same as an object of the application of the one or more objects of the application represented by the second one or more object representations, multiple objects of the application of the one or more objects of the application represented by the first one or more object representations are the same as multiple objects of the application of the one or more objects of the application represented by the second one or more object representations, all objects of the application of the one or more objects of the application represented by the first one or more object representations are the same as all objects of the application of the one or more objects of the application represented by the second one or more object representations, or all objects of the application of the one or more objects of the application represented by the first one or more object representations are different than all objects of the application of the one or more objects of the application represented by the second one or more object representations.

9. The system of claim **1**, wherein the first correlation is included in a first knowledge cell, and wherein the first knowledge cell is a data structure for storing, structuring, or organizing the first correlation.

10. A method implemented using a computing system that includes one or more processors, the method comprising:
accessing a memory that stores at least a first correlation including a first one or more object representations correlated with a first one or more instruction sets for operating a first avatar of an application, wherein the first one or more object representations represent one or more objects of the application;
generating or receiving a second one or more object representations, wherein the second one or more object representations represent one or more objects of the application;
determining the first one or more instruction sets for operating the first avatar of the application based on at least partial match between the second one or more object representations and the first one or more object representations; and
at least in response to the determining, executing the first one or more instruction sets for operating the first avatar of the application, wherein the first avatar of the application or a second avatar of the application performs one or more operations defined by the first one or more instruction sets for operating the first avatar of the application.

11. The system of claim **1**, wherein the first correlation is included in a knowledgebase, and wherein the knowledgebase further includes a third correlation including a third one or more object representations correlated with a third one or more instruction sets for operating the first avatar of the application, and wherein the third one or more object representations represent one or more objects of the application, and wherein at least a portion of the first correlation is learned in a learning process that includes operating the first avatar of the application at least partially by a user, and wherein at least a portion of the third correlation is learned in another learning process that includes operating the first avatar of the application at least partially by the user, and wherein the user is: a human user, or a non-human user.

12. The system of claim **1**, wherein the first correlation is included in a knowledgebase, and wherein the knowledgebase further includes a third correlation including a third one or more object representations correlated with a third one or more instruction sets for operating the first avatar of the application, and wherein the third one or more object representations represent one or more objects of the application, and wherein at least a portion of the first correlation is learned in a learning process that includes operating the first avatar of the application at least partially by a user, and wherein at least a portion of the third correlation is learned

in another learning process that includes operating the first avatar of the application at least partially by another user, and wherein the user is: a human user, or a non-human user, and wherein the another user is: a human user, or a non-human user.

13. The system of claim **1**, wherein the first correlation is included in a knowledgebase, and wherein the knowledgebase further includes a third correlation including a third one or more object representations correlated with a third one or more instruction sets for operating a third avatar of the application, and wherein the third one or more object representations represent one or more objects of the application, and wherein at least a portion of the first correlation is learned in a learning process that includes operating the first avatar of the application at least partially by a user, and wherein at least a portion of the third correlation is learned in another learning process that includes operating the third avatar of the application at least partially by: the user, or another user, and wherein the user is: a human user, or a non-human user, and wherein the another user is: a human user, or a non-human user.

14. The system of claim **1**, wherein the first correlation is included in a knowledgebase, and wherein the knowledgebase further includes a third correlation including a third one or more object representations correlated with a third one or more instruction sets for operating an avatar of another application, and wherein the third one or more object representations represent one or more objects of the another application.

15. The system of claim **1**, wherein the first one or more instruction sets for operating the first avatar of the application include one or more information about one or more states of: the first avatar of the application or a portion of the first avatar of the application.

16. The method system of claim **1**, wherein:

an element of the first correlation is deleted after the first correlation is generated,

an element of the first correlation is modified after the first correlation is generated,

an element of the first correlation is manipulated after the first correlation is generated, or

an element is inserted into the first correlation after the first correlation is generated.

17. The system of claim **1**, wherein the machine readable code, when executed by the one or more processors, causes the one or more processors to further perform at least:

modifying: the first one or more instruction sets for operating the first avatar of the application, or a copy of the first one or more instruction sets for operating the first avatar of the application, and wherein the executing the first one or more instruction sets for operating the first avatar of the application includes executing: the modified the first one or more instruction sets for operating the first avatar of the application, or the modified the copy of the first one or more instruction sets for operating the first avatar of the application, and wherein the performing, by the first avatar of the application or by the second avatar of the application, the one or more operations defined by the first one or more instruction sets for operating the first avatar of the application includes performing, by the first avatar of the application or by the second avatar of the application, one or more operations defined by: the modified the first one or more instruction sets for operating the first avatar of the application, or the modified the copy of the first one or more instruction sets for operating the first avatar of the application.

18. The system of claim **1**, wherein at least a portion of the first correlation is learned in a learning process that includes:

operating the first avatar of the application at least partially by: a human user, or a non-human user:

generating or receiving the first one or more object representations; and

obtaining or receiving the first one or more instruction sets for operating the first avatar of the application.

19. A system comprising:

a memory that stores at least a first correlation including a first one or more object representations correlated with a first one or more instruction sets for operating a first avatar of an application, wherein the first one or more object representations represent one or more objects of the application;

means for generating or receiving a second one or more object representations, wherein the second one or more object representations represent one or more objects of the application;

means for determining the first one or more instruction sets for operating the first avatar of the application based on at least partial match between the second one or more object representations and the first one or more object representations; and

means for executing, at least in response to the determining, the first one or more instruction sets for operating the first avatar of the application, wherein the first avatar of the application or a second avatar of the application performs one or more operations defined by the first one or more instruction sets for operating the first avatar of the application.

20. The system of claim **19**, wherein the means for generating or receiving the second one or more object representations includes one or more processors, and wherein the means for determining the first one or more instruction sets for operating the first avatar of the application based on the at least partial match between the second one or more object representations and the first one or more object representations includes one or more processors, and wherein the means for executing, at least in response to the determining, the first one or more instruction sets for operating the first avatar of the application includes one or more processors.

* * * * *

Exhibit D

US011663474B1

(12) **United States Patent**
Cosic

(10) **Patent No.: US 11,663,474 B1**
(45) **Date of Patent: *May 30, 2023**

(54) **ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION**

(71) Applicant: **Jasmin Cosic**, Miami, FL (US)

(72) Inventor: **Jasmin Cosic**, Miami, FL (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: **17/561,976**

(22) Filed: **Dec. 26, 2021**

Related U.S. Application Data

(63) Continuation of application No. 16/540,972, filed on Aug. 14, 2019, now Pat. No. 11,238,344, which is a continuation of application No. 15/340,991, filed on Nov. 2, 2016, now Pat. No. 10,452,974.

(51) **Int. Cl.**
G06N 3/08 (2023.01)
G06N 5/022 (2023.01)

(52) **U.S. Cl.**
CPC ***G06N 3/08*** (2013.01); ***G06N 5/022*** (2013.01)

(58) **Field of Classification Search**
CPC G06N 5/022; G06N 3/08
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,370,707 A 1/1983 Phillips et al.
4,730,315 A 3/1988 Saito et al.
4,860,203 A 8/1989 Corrigan et al.
5,560,011 A 9/1996 Uyama
5,602,982 A 2/1997 Judd et al.
5,874,834 A 2/1999 New
6,026,234 A 2/2000 Hanson et al.
6,088,731 A 7/2000 Kiraly et al.
6,106,299 A 8/2000 Ackermann et al.
6,126,330 A 10/2000 Knight
6,314,558 B1 11/2001 Angel et al.
6,408,262 B1 6/2002 Leerberg et al.
6,643,842 B2 11/2003 Angel et al.
6,728,689 B1 4/2004 Drissi et al.
6,735,632 B1 5/2004 Kiraly et al.
6,754,631 B1 6/2004 Din
6,801,912 B2 10/2004 Moskowitz et al.
6,842,877 B2 1/2005 Robarts et al.
6,915,105 B2 7/2005 Masuda
6,973,446 B2 12/2005 Mamitsuka et al.
6,996,758 B1 2/2006 Herron et al.
(Continued)

OTHER PUBLICATIONS

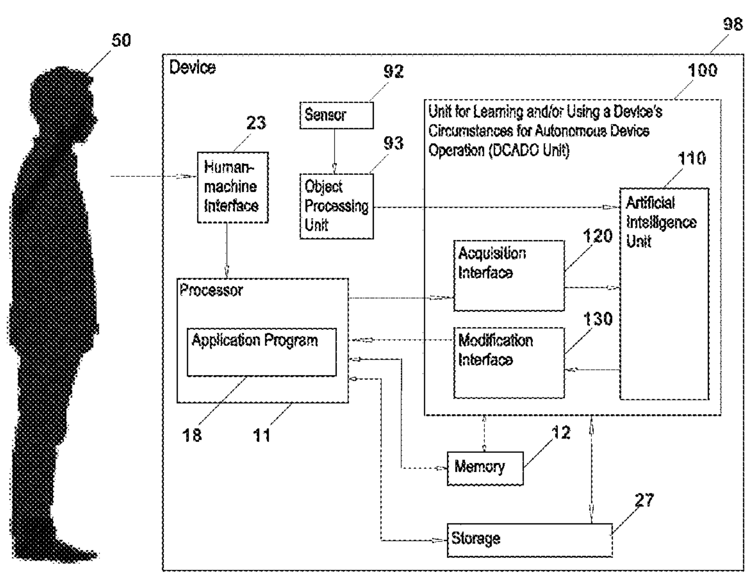
Tracing (software), retrieved from <URL: http://wikipedia.com> on Jan. 10, 2014, 3 pages.
(Continued)

Primary Examiner — Santiago Garcia
(74) *Attorney, Agent, or Firm* — Imperium Patent Works LLP

(57) **ABSTRACT**

Aspects of the disclosure generally relate to computing enabled devices and/or systems, and may be generally directed to devices, systems, methods, and/or applications for learning a device's operation in various circumstances, storing this knowledge in a knowledgebase (i.e. neural network, graph, sequences, etc.), and enabling autonomous operation of the device.

84 Claims, 40 Drawing Sheets



(56) **References Cited**

U.S. PATENT DOCUMENTS

7,017,153	B2	3/2006	Gouriou et al.
7,052,277	B2	5/2006	Kellman
7,113,946	B2	9/2006	Cosic
7,117,225	B2	10/2006	Cosic
7,222,127	B1	5/2007	Bem et al.
7,240,335	B2	7/2007	Angel et al.
7,249,349	B2	7/2007	Hundt et al.
7,424,705	B2	9/2008	Lewis et al.
7,478,371	B1	1/2009	Gove
7,484,205	B2	1/2009	Venkatapathy
7,565,340	B2	7/2009	Herlocker et al.
7,721,218	B2	5/2010	Awe et al.
7,765,537	B2	7/2010	Havin et al.
7,797,259	B2	9/2010	Jiang et al.
7,840,060	B2	11/2010	Podilchuk
7,925,984	B2	4/2011	Awe et al.
7,987,144	B1	7/2011	Drissi et al.
8,005,828	B2	8/2011	Buchner et al.
8,019,699	B2	9/2011	Baxter
8,078,556	B2	12/2011	Adi et al.
8,090,669	B2	1/2012	Shahani et al.
8,137,112	B2	3/2012	Woolf et al.
8,166,463	B2	4/2012	Gill et al.
8,195,674	B1	6/2012	Bem et al.
8,196,119	B2	6/2012	Gill et al.
8,244,730	B2	8/2012	Gupta
8,251,704	B2	8/2012	Woolf et al.
8,261,199	B2	9/2012	Cradick et al.
8,266,608	B2	9/2012	Hecht et al.
8,335,805	B2	12/2012	Cosic
8,356,026	B2	1/2013	Heimendinger
8,364,612	B2	1/2013	Van Gael et al.
8,386,401	B2	2/2013	Virkar et al.
8,397,227	B2	3/2013	Fan et al.
8,417,740	B2	4/2013	Cosic
8,464,225	B2	6/2013	Greifeneder
8,549,359	B2	10/2013	Zheng
8,572,035	B2	10/2013	Cosic
8,589,414	B2	11/2013	Waite et al.
8,595,154	B2	11/2013	Breckenridge et al.
8,655,260	B2	2/2014	Yueh et al.
8,655,900	B2	2/2014	Cosic
8,667,472	B1	3/2014	Molinari
8,996,432	B1	3/2015	Fu
9,047,324	B2	6/2015	Cosic
9,158,813	B2	10/2015	Ntoulas et al.
9,268,454	B2	2/2016	Hamilton, II et al.
9,282,309	B1	3/2016	Cosic
9,298,749	B2	3/2016	Cosic
9,305,216	B1	4/2016	Mishra
9,325,314	B2	4/2016	Shin
9,367,806	B1	6/2016	Cosic
9,443,192	B1	9/2016	Cosic
9,488,984	B1	11/2016	Williams et al.
10,162,740	B1	12/2018	Setty et al.
2003/0026588	A1	2/2003	Elder et al.
2003/0065662	A1	4/2003	Cosic
2003/0122578	A1	7/2003	Masui et al.
2004/0117771	A1	6/2004	Venkatapathy
2004/0194017	A1	9/2004	Cosic
2004/0249774	A1	12/2004	Caid et al.
2004/0267521	A1	12/2004	Cutler et al.
2005/0149517	A1	7/2005	Cosic
2005/0149542	A1	7/2005	Cosic
2005/0240412	A1	10/2005	Fujita
2005/0245303	A1	11/2005	Graepel et al.
2005/0289105	A1	12/2005	Cosic
2006/0047612	A1	3/2006	Stanley et al.
2006/0184410	A1	8/2006	Ramamurthy et al.
2006/0190930	A1	8/2006	Hecht et al.
2006/0265406	A1	11/2006	Chkodrov et al.
2007/0006159	A1	1/2007	Hecht et al.
2007/0050606	A1	3/2007	Ferren et al.
2007/0050719	A1	3/2007	Lui et al.
2007/0058856	A1	3/2007	Boregowda et al.
2007/0061735	A1*	3/2007	Hoffberg G06V 40/103 715/744
2007/0106633	A1	5/2007	Reiner
2008/0144893	A1	6/2008	Guo et al.
2008/0215508	A1	9/2008	Hanneman et al.
2008/0254429	A1	10/2008	Woolf et al.
2008/0281764	A1	11/2008	Baxter
2008/0288259	A1	11/2008	Chambers et al.
2009/0044113	A1	2/2009	Jones et al.
2009/0067727	A1	3/2009	Hirohata
2009/0110061	A1	4/2009	Lee et al.
2009/0131152	A1	5/2009	Busse
2009/0136095	A1	5/2009	Marcon et al.
2009/0141969	A1	6/2009	Yu et al.
2009/0222388	A1	9/2009	Hua et al.
2009/0287643	A1	11/2009	Corville et al.
2009/0288001	A1*	11/2009	Hamilton, II G06F 9/453 715/706
2009/0324010	A1	12/2009	Hou
2010/0023541	A1	1/2010	Cosic
2010/0033780	A1	2/2010	Gitter
2010/0063949	A1	3/2010	Eaton et al.
2010/0082536	A1	4/2010	Cosic
2010/0114746	A1	5/2010	Bobbitt et al.
2010/0138370	A1	6/2010	Wu et al.
2010/0241595	A1	9/2010	Felsher
2010/0278420	A1	11/2010	Shet et al.
2011/0007079	A1	1/2011	Perez et al.
2011/0030031	A1	2/2011	Lussier et al.
2011/0085734	A1	4/2011	Berg et al.
2011/0218672	A1	9/2011	Maisonnier et al.
2011/0270794	A1	11/2011	Drory et al.
2012/0150773	A1	6/2012	DiCorpo et al.
2012/0167057	A1	6/2012	Schmich et al.
2012/0192096	A1	7/2012	Bowman et al.
2012/0284026	A1	11/2012	Cardillo et al.
2012/0290347	A1	11/2012	Elazouni et al.
2013/0007532	A1	1/2013	Miller et al.
2013/0047049	A1	2/2013	Chen et al.
2013/0156345	A1	6/2013	Shmunk
2013/0159021	A1	6/2013	Felsher
2013/0218932	A1	8/2013	Cosic
2013/0226974	A1	8/2013	Cosic
2013/0238533	A1	9/2013	Virkar et al.
2013/0278501	A1	10/2013	Bulzacki
2013/0278631	A1	10/2013	Border et al.
2013/0329946	A1	12/2013	Archibald et al.
2014/0052717	A1	2/2014	Franks et al.
2014/0075249	A1	3/2014	Sato et al.
2014/0161250	A1	6/2014	Leeds et al.
2014/0177946	A1	6/2014	Lim et al.
2014/0207580	A1	6/2014	Minnis et al.
2014/0211988	A1	7/2014	Fan et al.
2014/0278754	A1	9/2014	Cronin et al.
2015/0006171	A1	1/2015	Westby et al.
2015/0023186	A1*	1/2015	Vasseur H04L 43/103 370/252
2015/0023205	A1*	1/2015	Vasseur H04L 45/02 370/254
2015/0039304	A1	2/2015	Wein
2015/0055821	A1	2/2015	Fotland
2015/0089081	A1*	3/2015	Thubert H04W 40/30 709/239
2015/0264306	A1	9/2015	Marilly et al.
2015/0269156	A1	9/2015	Awadallah et al.
2015/0269415	A1	9/2015	Gelbman
2015/0310041	A1	10/2015	Kier et al.
2015/0324685	A1*	11/2015	Bohn G06N 3/04 706/15
2015/0339213	A1*	11/2015	Lee G06F 3/0484 717/125
2016/0063118	A1	3/2016	Campbell et al.
2016/0077166	A1	3/2016	Morozov et al.
2016/0140999	A1	5/2016	Cosic
2016/0142650	A1	5/2016	Cosic
2016/0167226	A1	6/2016	Schnittman
2016/0243701	A1	8/2016	Gildert et al.
2016/0246819	A1	8/2016	Cosic
2016/0246850	A1	8/2016	Cosic

(56) **References Cited**

U.S. PATENT DOCUMENTS

2016/0246868	A1	8/2016	Cosic	
2016/0274187	A1	9/2016	Menon et al.	
2016/0292185	A1	10/2016	Cosic	
2016/0306806	A1*	10/2016	Fackler	G06T 11/206
2016/0328480	A1*	11/2016	Owens	G06F 16/9535
2017/0161958	A1	6/2017	Eilat	
2018/0017791	A1*	1/2018	Beckman	H04N 23/687
2018/0060144	A1*	3/2018	Capobianco	G06F 1/1613
2018/0068564	A1	3/2018	Tanigawa et al.	
2018/0096261	A1*	4/2018	Chu	G06N 20/00
2018/0101146	A1	4/2018	Hariharan et al.	
2018/0120813	A1	5/2018	Coffman et al.	
2018/0137403	A1*	5/2018	Kaur	G06F 11/1608
2018/0174207	A1*	6/2018	Potucek	G06Q 50/08
2018/0375720	A1*	12/2018	Yang	H04L 41/084
2019/0028545	A1*	1/2019	Yang	H04W 4/70
2019/0034765	A1	1/2019	Kaehler et al.	
2019/0096503	A1	3/2019	Chakravarty et al.	
2019/0107839	A1	4/2019	Parashar et al.	
2019/0213446	A1*	7/2019	Tsou	G06F 18/2148

OTHER PUBLICATIONS

Tree (data structure), retrieved from <URL: http://wikipedia.com> on Jun. 24, 2014, 6 pages.

PTRACE(2), retrieved from <URL: http://unixhelp.ed.ac.uk/CGI/man-cgi?ptrace> on Mar. 19, 2014, 5 pages.

Wevtutil, retrieved from <URL: http://technet.microsoft.com/en-us/library/cc732848(d=default,I=en-us,v=ws.11).aspx> on Apr. 28, 2014, 5 pages.

Intel Processor Trace, retrieved from <URL: https://software.intel.com/en-us/blogs/2013/09/18/processor-tracing> on Apr. 28, 2014, 3 pages.

Younghoon Jung, Java Dynamics Reflection and a lot more, Oct. 10, 2012, 55 pages, Columbia University.

Amitabh Srivastava, Alan Eustace, ATOM A System for Building Customized Program Analysis Tools, May 3, 2004, 12 pages.

Mathew Smithson, Kapil Anand, Aparna Kotha, Khaled Elwazeer, Nathan Giles, Rajeev Barua, Binary Rewriting without Relocation Information, Nov. 10, 2010, 11 pages, University of Maryland.

Marek Olszewski, Keir Mierte, Adam Czajkowski, Angela Demle Brown, JIT Instrumentation—A Novel Approach to Dynamically Instrument Operating Systems, Feb. 12, 2007, 14 pages, University of Toronto.

Chen et al. Case-Based Reasoning System and Artificial Neural Networks: A Review Neural Comput & Applic (2001) 10: pp. 264-276, 13 pages.

John J. Grefenstette, Connie Loggia Ramsey, Alan C. Schultz, Learning Sequential Decision Rules Using Simulation Models and Competition, 1990, Navy Center for Applied Research in Artificial Intelligence, Naval Research Laboratory, Washington, DC, 27 pages.

Alan C. Schultz, John J. Grefenstette, Using a Genetic Algorithm to Learn Behaviors for Autonomous Vehicles, 1992, Navy Center for Applied Research in Artificial Intelligence, Naval Research Laboratory, Washington, DC, 12 pages.

Koppula et al., "Anticipating human activities using object affordances for reactive robotic response", IEEE TRAMI 2016, published May 5, 2015, 16 pages.

Orme, "System design tips for entry level smartphones—part 3", found online at "https://community.arm.com/processors/b/blog/posts/system-design-tips-for-entry-level-smartphones—part-3", Oct. 21, 2013, 12 pages.

Nowostawski, Mariusz et al., "Dynamic Demes Parallel Genetic Algorithm", May 13, 1999, 6 pages.

Stack Overflow, How can I quantify difference between two images?, accessed Aug. 2, 2017 at https://stackoverflow.com/questions/189943/how-can-i-quantify-difference-between-two-images, 8 pages.

Skolicki, Zbigniew et al., "The Influence of Migration Sizes and Intervals on Island Models", Jun. 29, 2005, 8 pages.

Vahid Lari, et al., "Decentralized dynamic resource management support for massively parallel processor arrays", Sep. 11, 2011, 8 pages.

Vahid Lari, et al., "Distributed resource reservation in massively parallel processor arrays", May 16, 2011, 4 pages.

Mohsen Hayati and Yazdan Shirvany, "Artificial Neural Network Approach for Short Term Load Forecasting for Illam Region", Jan. 2007, 5 pages.

Hasim Sak, Andrew Senior, and Francoise Beaufays, "Long Short-Term Memory Recurrent Neural Network Architectures for Large Scale Acoustic Modeling", Jan. 2014, 5 pages.

Jorg Walter and Klaus Schulten, "Implementation of self-organizing neural networks for visuo-motor control of an industrial robot", Jan. 1993, 10 pages.

Heikki Hyotyniemi and Aamo Lehtola, "A Universal Relation Database Interface for Knowledge Based Systems", Apr. 1991, 5 pages.

Mrissa, Michael, et al. "An avatar architecture for the web of things." IEEE Internet Computing 19.2 (2015): 30-38., 9 pages.

Luck, Michael, and Ruth Aylett. "Applying artificial intelligence to virtual reality: Intelligent virtual environments." Applied Artificial Intelligence 14.1 (2000): 3-32., 30 pages.

Terdjimi, Mehdi, et al. "An avatar-based adaptation workflow for the web of things." Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), 2016 IEEE 25th International Conference on. IEEE, 2016., 6 pages.

Bogdanovych, Anton, et al. "Authentic interactive reenactment of cultural heritage with 3D virtual worlds and artificial intelligence." Applied Artificial Intelligence 24.6 (2010): 617-647., 32 pages.

Hernandez, Marco E. Perez, and Stephan Reiff-Marganiec. "Autonomous and self controlling smart objects for the future internet." Future internet of things and cloud (FiCloud), 2015 3rd international conference on. IEEE, 2015., 8 pages.

Medini et al., "Building a Web of Things with Avatars", Managing the Web of Things (2017), 30 pages.

About Event Tracing, retrieved from <URL: http://msdn.microsoft.com/en-us/library/aa363668(d=default,l=en-us,v=vs.85).aspx> on Jan. 12, 2014, 2 pages.

Add dynamic Java code to your application, retrieved from <URL: http://www.javaworld.com/article/2071777/design-patterns/add-dynamic-java-code-to-your-application.html> on May 13, 2014, 6 pages.

BCEL, retrieved from <URL: http://commons.apache.org> on May 13, 2014, 2 pages.

Artificial intelligence, retrieved from <URL: http://wikipedia.com> on May 21, 2014, 28 pages.

Artificial neural network, retrieved from <URL: http://wikipedia.com> on May 21, 2014, 12 pages.

Branch (computer science), retrieved from <URL: http://wikipedia.com> on May 6, 2014, 2 pages.

Branch trace, retrieved from <URL: http://wikipedia.com> on May 28, 2014, 2 pages.

C Function Call Conventions and the Stack, retrieved from <URL: http://www.csee.umbc.edu/~chang/cs313.s02/stack.shtml> on Jan. 11, 2014, 4 pages.

Call stack, retrieved from <URL: http://wikipedia.com> on Jan. 11, 2014, 5 pages.

Program compilation and execution flow, retrieved from <URL: http://cs.stackexchange.com/questions/6187/program-compilation-and-execution-flow> on Jan. 9, 2014, 2 pages.

Control unit, retrieved from <URL: http://wikipedia.com> on May 25, 2014, 2 pages.

Create dynamic applications with javax.tools, retrieved from <URL: http://www.ibm.com/developerworks/library/j-jcomp/> on May 13, 2014, 11 pages.

Creating a Debugging and Profiling Agent with JVMTI, retrieved from <URL: http://www.oracle.com/technetwork/articles/java/jvmti-136367.html> on Jan. 11, 2014, 18 pages.

Decision tree learning, retrieved from <URL: http://wikipedia.com> on Jun. 24, 2014, 5 pages.

Dynamic Bytecode Instrumentation, retrieved from <URL: http://www.drdobbs.com/tools/dynamic-bytecode-instrumentation/184406433> on Apr. 26, 2014, 6 pages.

(56) **References Cited**

OTHER PUBLICATIONS

Dynamic programming language, retrieved from <URL: http://wikipedia.com> on May 7, 2014, 4 pages.
Dynamic recompilation, retrieved from <URL: http://wikipedia.com> on May 19, 2014, 4 pages.
Eval(), retrieved from <URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/eval> on May 7, 2014, 13 pages.
Function, retrieved from <URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Function> on May 7, 2014, 11 pages.
FunctionEnter2 Function, retrieved from <URL: https://msdn.microsoft.com/en-us/library/aa964981(d=default,l=en-us,v=vs.110).aspx> on Mar. 19, 2014, 2 pages.
FunctionLeave2 Function, retrieved from <URL: https://msdn.microsoft.com/en-us/library/aa964942(d=default,l=en-us,v=vs.110).aspx> on Mar. 19, 2014, 1 pages.
Heuristic, retrieved from <URL: http://wikipedia.com> on May 22, 2014, 5 pages.
How to: Configure Trace Switches, retrieved from <URL: https://msdn.microsoft.com/en-us/library/vstudio/t06xyy08(d=default,l=en-us,v=vs.100).aspx> on Mar. 19, 2014, 3 pages.
Indirect branch, retrieved from <URL: http://wikipedia.com> on May 5, 2014, 1 pages.
Instruction cycle, retrieved from <URL: http://wikipedia.com> on Apr. 30, 2014, 2 pages.
Instruction set, retrieved from <URL: http://wikipedia.com> on Apr. 29, 2014, 6 pages.
Interrupt, retrieved from <URL: http://wikipedia.com> on May 15, 2014, 5 pages.
Introduction to Instrumentation and Tracing, retrieved from <URL: http://msdn.microsoft.com/en-us/library/aa983649(d=default,l=en-us,v=vs.71).aspx> on Jan. 10, 2014, 2 pages.
Java bytecode, retrieved from <URL: http://wikipedia.com> on May 27, 2014, 4 pages.
Package java.util.logging, retrieved from <URL: http://docs.oracle.com/javase/1.5.0/docs/api/java/util/logging/package-summary.html> on Mar. 19, 2014, 2 pages.
Javac—Java programming language compiler, retrieved from <URL: http://docs.oracle.com/javase/7/docs/technotes/tools/windows/javac.html> on May 13, 2014, 12 pages.
Interface JavaCompiler, retrieved from <URL: http://wikipedia.com> on May 13, 2014, 4 pages.
Javassist, retrieved from <URL: http://wikipedia.com> on May 13, 2014, 2 pages.
Just-in-time compilation, retrieved from <URL: http://wikipedia.com> on May 19, 2014, 5 pages.
JVM Tool Interface, retrieved from <URL: http://docs.oracle.com/javase/7/docs/platform/jvmti/jvmti.html> on Apr. 26, 2014, 127 pages.
Machine code, retrieved from <URL: http://wikipedia.com> on Apr. 30, 2014, 3 pages.
Metaprogramming, retrieved from <URL: http://wikipedia.com> on May 8, 2014, 3 pages.
Microsoft.VisualBasic.Logging Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/ms128122(d=default,l=en-us,v=vs.110).aspx> on Jan. 13, 2014, 1 pages.
Pin (computer program), retrieved from <URL: http://wikipedia.com> on May 6, 2014, 3 pages.
Processor register, retrieved from <URL: http://wikipedia.com> on Apr. 29, 2014, 4 pages.
Profiling (computer programming), retrieved from <URL: http://wikipedia.com> on Jan. 10, 2014, 4 pages.
Profiling Overview, retrieved from <URL: http://msdn.microsoft.com/en-us/library/bb384493(d=default,l=en-us,v=vs.110).aspx> on Jan. 12, 2014, 6 pages.
Program counter, retrieved from <URL: http://wikipedia.com> on Apr. 24, 2014, 2 pages.
Ptrace, retrieved from <URL: http://wikipedia.com> on Apr. 24, 2014, 2 pages.
Reflection (computer programming), retrieved from <URL: http://wikipedia.com> on May 7, 2014, 5 pages.
Class Runtime, retrieved from <URL: http://docs.oracle.com/javase/7/docs/api/java/lang/Runtime.html> on Jan. 10, 2014, 10 pages.
Self-modifying code, retrieved from <URL: http://wikipedia.com> on Jan. 12, 2014, 6 pages.
Subjective logic, retrieved from <URL: http://wikipedia.com> on May 22, 2014, 5 pages.
System.CodeDom.Compiler Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/z6b99ydt(d=default,l=en-us,v=vs.110).aspx> on May 13, 2014, 2 pages.
System.Reflection.Emit Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/xd5fw18y(d=default,l=en-us,v=vs.110).aspx> on May 13, 2014, 2 pages.
Zhang, Yiying, Lei Guo, and Nicolas D. Georganas. "AGILE: An architecture for agent-based collaborative and interactive virtual environments." Proc. Workshop on Application Virtual Reality Technologies for Future Telecommunication System, IEEE Globecom '2000 Conference. 2000. (Year: 2000), 9 pages.
Eladhari, Mirjam P., "Semi-Autonomous Avatars in Virtual Game Worlds," Pre-conference to the ECREA 2010—3rd European Communication Conference, Avatars and Humans. Representing Users in Digital Games, Hamburg, Germany, Oct. 2010, Available from: Jan. 21, 2012 (Year: 2012), 8 pages.
Matthew Rimmer, "Tesla Motors' Open Source Revolution: Intellectual Property and the Carbon Crisis," Medium (Jun. 15, 2014), 13 pages.
Keith Barry, "Guide to Lane Departure Warning and Lane Keeping Assist," Consumer Reports (Jun. 28, 2017), 13 pages.
The Tesla Team, "Dual Motor Model S and Autopilot," Tesla Blog (Oct. 10, 2014), 3 pages.
Gabe Nelson, "Tesla enables 'autopilot' for the Model S, promises a world without steering wheels," Automotive News (Oct. 15, 2015), 3 pages.
Chris Davies, "Tesla Enhanced Autopilot: What 8.1 is (and isn't) and 2016 will bring," Slash Gear (Dec. 22, 2016), 6 pages.
Andrew J. Hawkins, "Tesla has been working on a backup plan in case its self-driving promises fail," The Verge (Aug. 9, 2017), 5 pages.
Jordan Golson, "Tesla's Self-Driving Care Hardware Will Run You $8,000," The Verge (Oct. 20, 2016), 3 pages.
Kirsten Korosec, "Elon Musk Says Tesla Vehicles Will Drive Themselves in Two Years," Fortune (Dec. 21, 2015), 7 pages.
XAutoworld, "Transcript: Elon Musk's Autopilot 2.0 Conference Call," XAutoworld (Nov. 20, 2016), 17 pages.
About | OpenCV, retrieved from <URL: http://opencv.org/about.html> on Dec. 13, 2014, 1 pages.
Animetrics, Inc. | 3D Facial Recognition, retrieved from <URL: http//animetrics.com/> on Dec. 13, 2014, 2 pages.
Bag-of-words model, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 2 pages.
Beier-Neely morphing algorithm, retrieved from <URL: http//wikipedia.com> on Nov. 19, 2015, 1 pages.
Computer vision, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 7 pages.
Convolutional neural network, retrieved from <URL: http://wikipedia.com> on Nov. 11, 2015, 5 pages.
Decimation (signal processing), retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 3 pages.
Digital image processing, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 3 pages.
Dissolve (filmmaking), retrieved from <URL: http://wikipedia.com> on Nov. 11, 2015, 2 pages.
Facial recognition system, retrieved from <URL: http://wikipedia.com> on Nov. 11, 2015, 6 pages.
Feature detection (computer vision), retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 3 pages.
Feature extraction, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 2 pages.
Gesture recognition, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 5 pages.
GrabCut, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 1 pages.

(56) **References Cited**

OTHER PUBLICATIONS

Image processing, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 2 pages.
Image segmentation, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 12 pages.
Image warping, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 2 pages.
Inbetweening, retrieved from <URL: http://wikipedia.com> on Nov. 11, 2015, 2 pages.
Interpolation, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 5 pages.
Language model, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 4 pages.
List of speech recognition software, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 4 pages.
Livewire Segmentation Technique, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 2 pages.
Morphing, retrieved from <URL: http://wikipedia.com> on Nov. 11, 2015, 2 pages.
Motion estimation, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 2 pages.
Muse: Face Morph || Mesh Warping, retrieved from <URL: http://alexwolfe.blogspot.com/2011/10/face-morph-mesh-warping.html> on Nov. 19, 2015, 2 pages.
N-gram, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 5 pages.
Optical flow, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 4 pages.
Outline of object recognition, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 7 pages.
Phonotactics, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 3 pages.
Recurrent neural network, retrieved from <URL: http://wikipedia.com> on Nov. 11, 2015, 5 pages.
Sample rate conversion, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 2 pages.
Simple interactive object extraction, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 2 pages.
Speech recognition, retrieved from <URL: http://wikipedia.com> on Nov. 11, 2015, 12 pages.
Speech segmentation, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 4 pages.
Welcome to recognize-speech.com, retrieved from <URL: http://recognize-speech.com/> on Oct. 18, 2015, 1 pages.
Introduction Speech, retrieved from <URL: http://recognize-speech.com/speech> on Oct. 18, 2015, 1 pages.
Preprocessing, retrieved from <URL: http://recognize-speech.com/preprocessing> on Oct. 18, 2015, 4 pages.
Feature Extraction, retrieved from <URL: http://recognize-speech.com/feature-extraction> on Oct. 18, 2015, 3 pages.
Acoustic model, retrieved from <URL: http://recognize-speech.com/acoustic-model> on Oct. 18, 2015, 2 pages.
Video content analysis, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 2 pages.
Video tracking, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 2 pages.
Andrej Karpathy, Thomas Leung, George Toderici, Rahul Sukthankar, Sanketh Shetty, Li Fei-Fei, Large-scale Video Classification with Convolutional Neural Networks, Apr. 14, 2014, 8 pages, Stanford University.
Karen Simonyan, Andrew Zisserman, Two-Stream Convolutional Networks for Action Recognition in Videos, Nov. 13, 2014, 11 pages, University of Oxford.

* cited by examiner

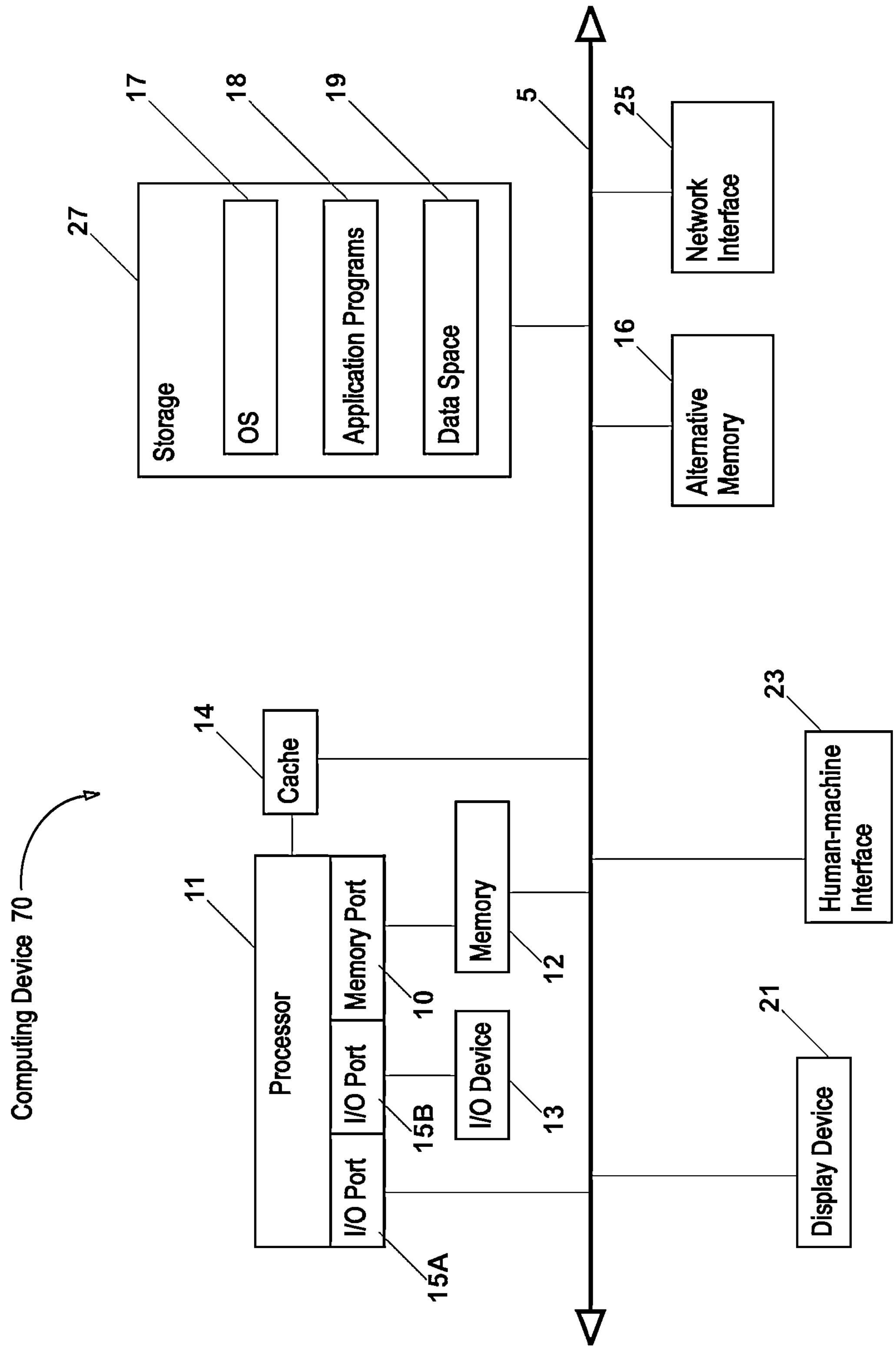

Computing Device 70
Storage
OS
Application Programs
Data Space
27
17
18
19
5
25
Network Interface
16
Alternative Memory
14
Cache
11
Processor
Memory Port
I/O Port
I/O Port
10
15B
15A
Memory
12
I/O Device
13
23
Human-machine Interface
21
Display Device

FIG. 1

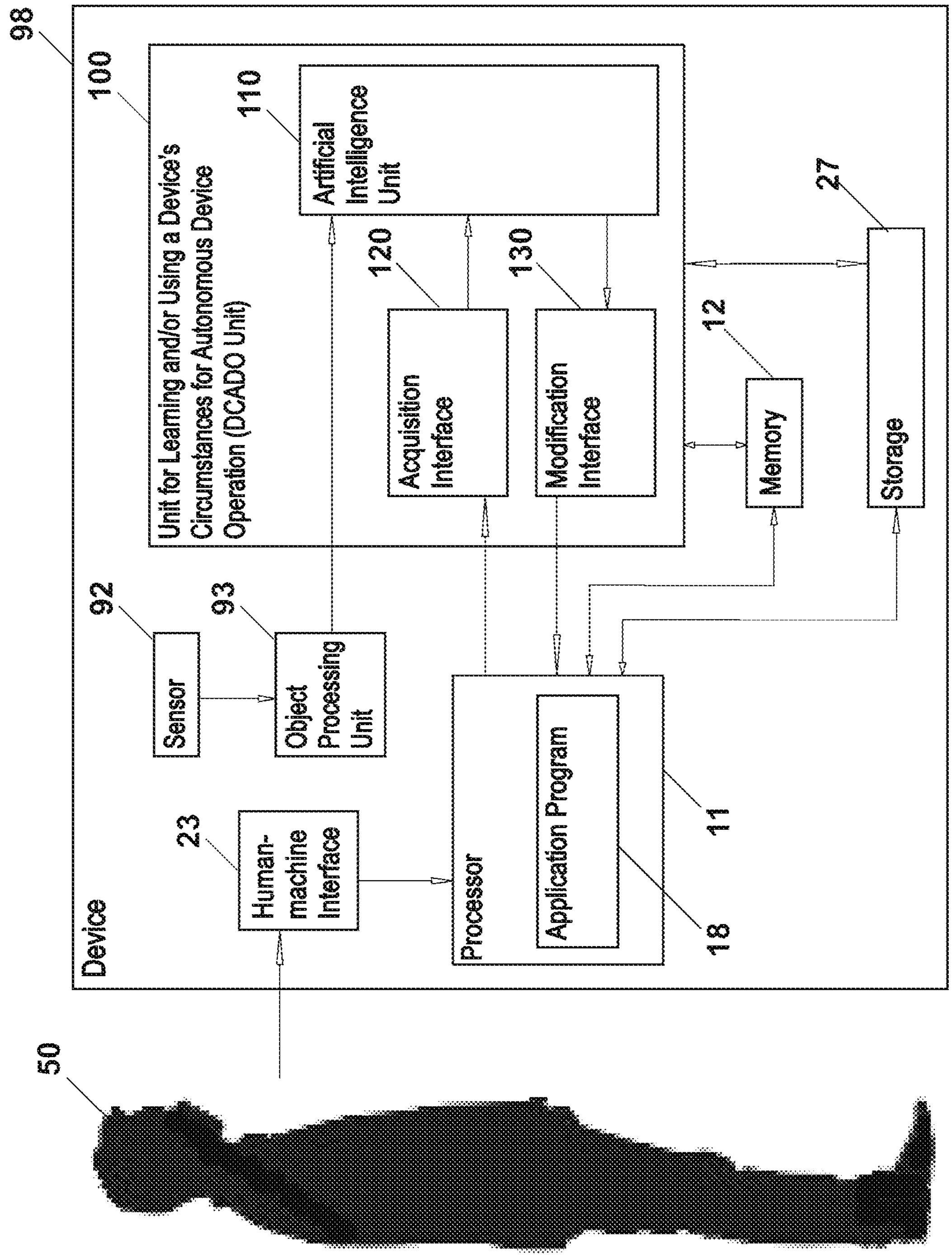

98
100
Unit for Learning and/or Using a Device's Circumstances for Autonomous Device Operation (DCADO Unit)
110
Artificial Intelligence Unit
120
Acquisition Interface
130
Modification Interface
12
Memory
27
Storage
92
Sensor
93
Object Processing Unit
23
Human-machine Interface
Device
Processor
Application Program
11
18
50

FIG. 2

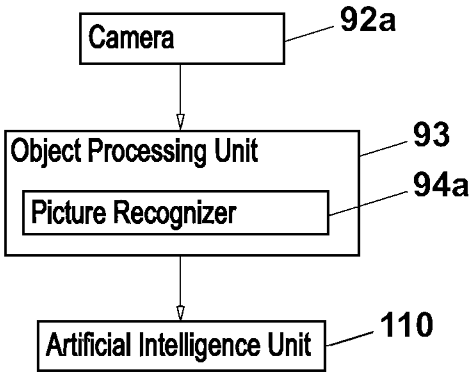


FIG. 3A

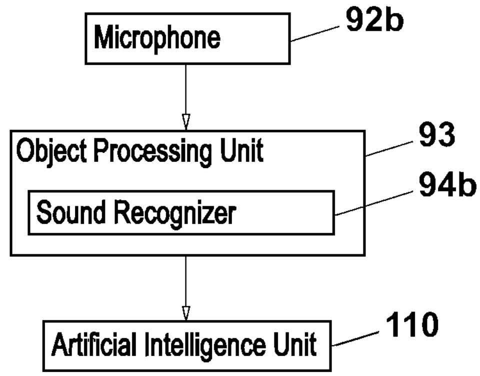


FIG. 3B

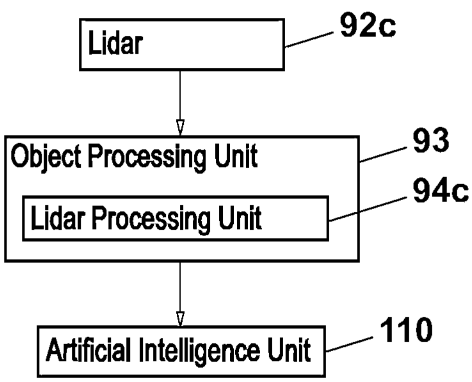


FIG. 3C

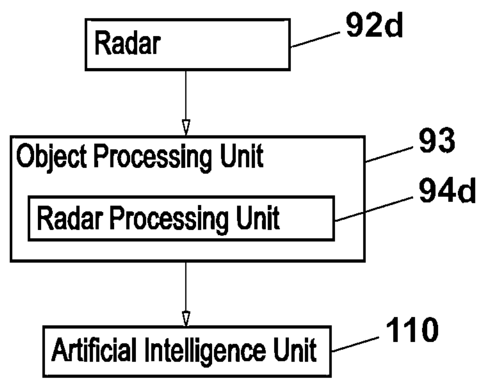


FIG. 3D

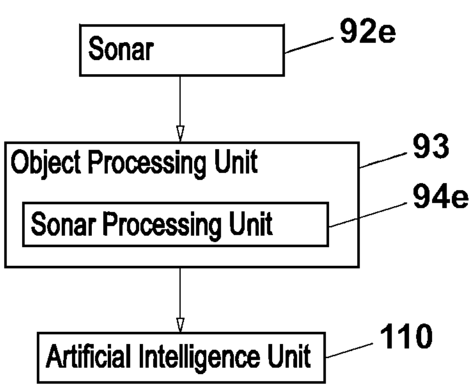


FIG. 3E

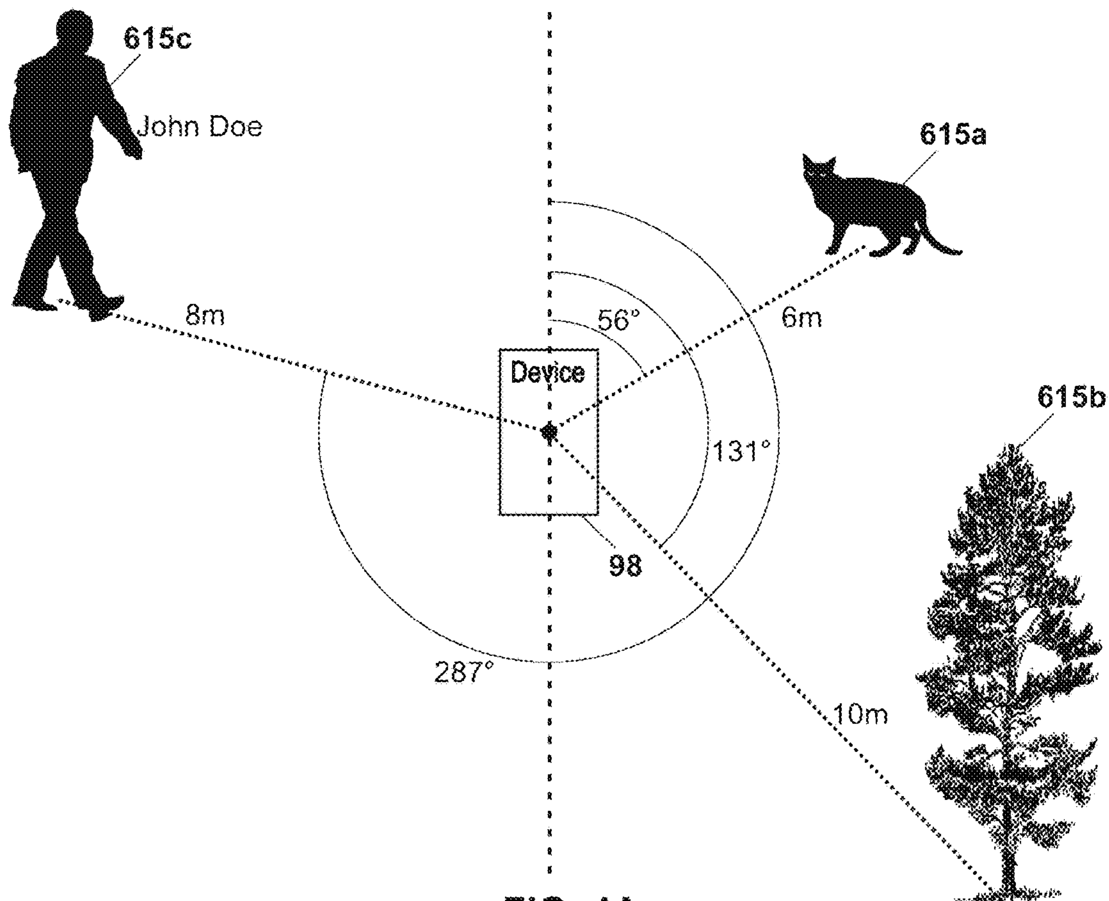

615c
John Doe
615a
8m
56°
6m
Device
615b
131°
98
287°
10m

FIG. 4A

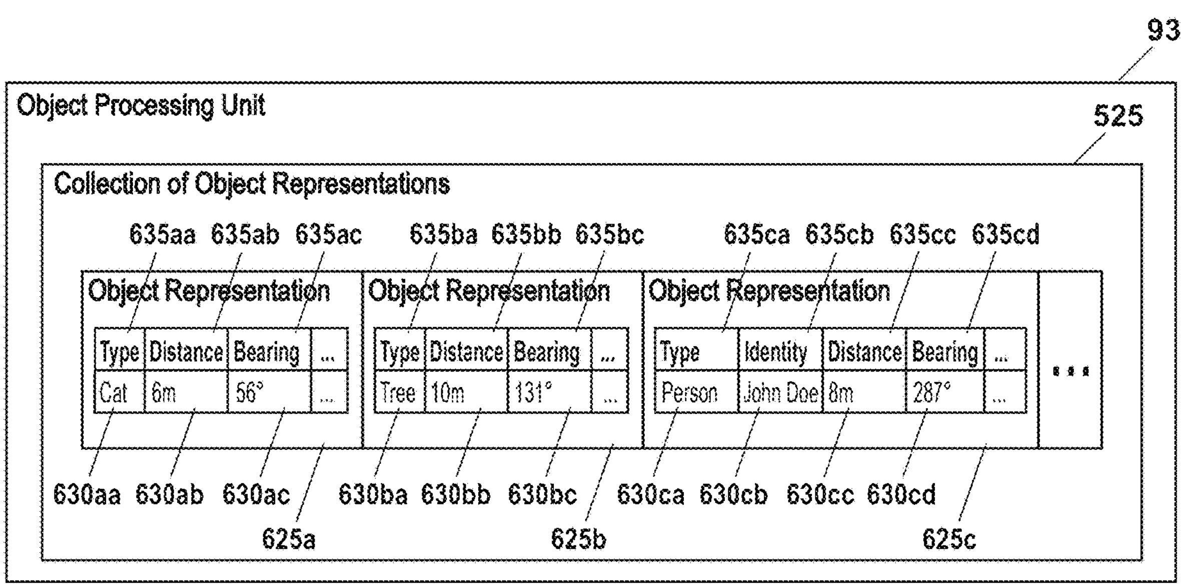

93
Object Processing Unit
525
Collection of Object Representations
635aa 635ab 635ac
635ba 635bb 635bc
635ca 635cb 635cc 635cd
Object Representation
Type Distance Bearing ...
Cat 6m 56° ...
Object Representation
Type Distance Bearing ...
Tree 10m 131° ...
Object Representation
Type Identity Distance Bearing ...
Person John Doe 8m 287° ...
...
630aa 630ab 630ac
625a
630ba 630bb 630bc
625b
630ca 630cb 630cc 630cd
625c

FIG. 4B

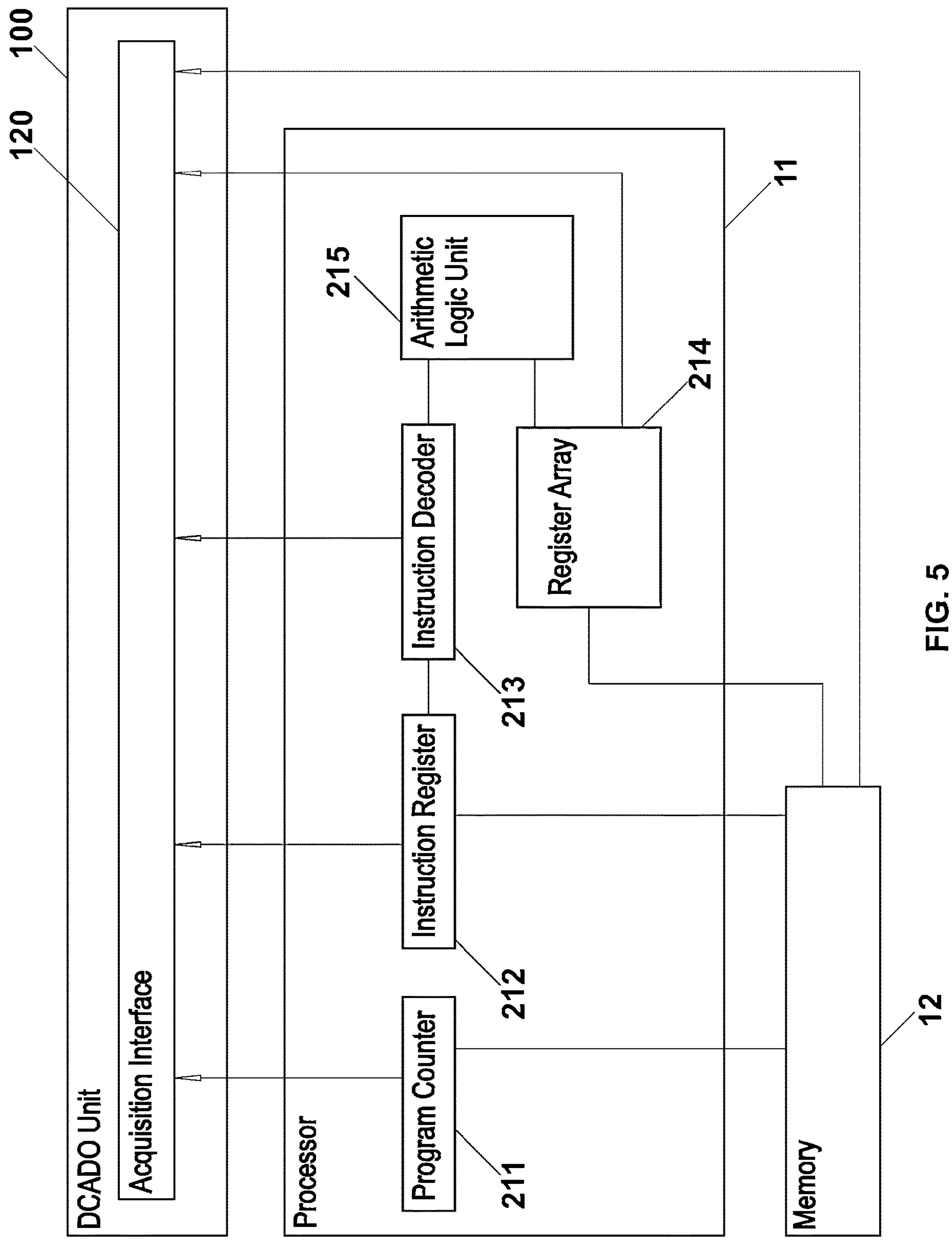

100
120
DCADO Unit
Acquisition Interface
Processor
11
Program Counter
211
Instruction Register
212
Instruction Decoder
213
Register Array
214
Arithmetic Logic Unit
215
Memory
12

FIG. 5

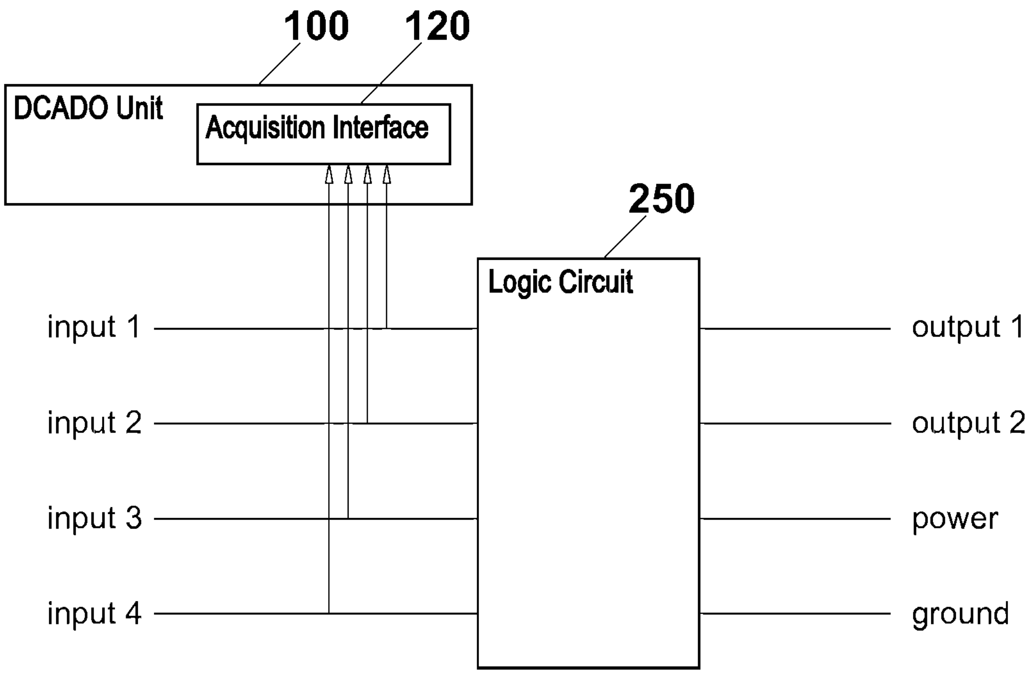

100
120
DCADO Unit
Acquisition Interface
250
Logic Circuit
input 1
input 2
input 3
input 4
output 1
output 2
power
ground

FIG. 6A

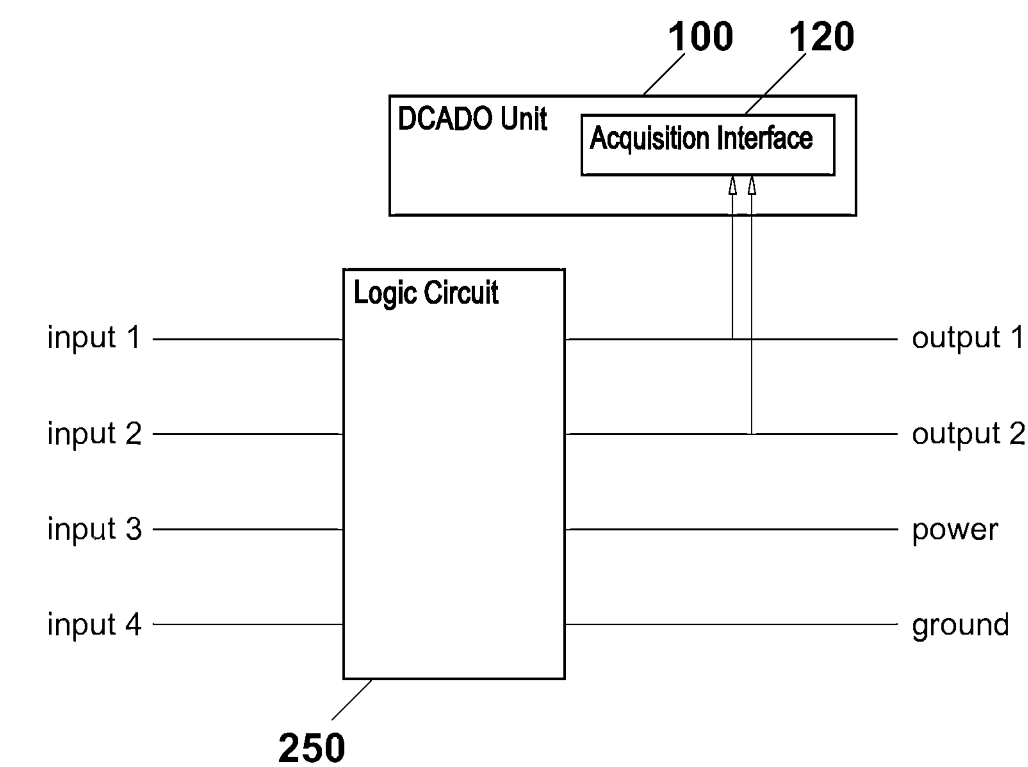

100
120
DCADO Unit
Acquisition Interface
Logic Circuit
input 1
input 2
input 3
input 4
output 1
output 2
power
ground
250

FIG. 6B

526

Instruction Set

Function (Parameter1, Parameter2, Parameter3, ...)

FIG. 7A

526

Instruction Set

SELECT Col1, Col2 ... FROM Tbl1, Tbl2 ... ;

FIG. 7B

526

Instruction Set

iconst_Value

FIG. 7C

526

Instruction Set

add Register1 Register2

FIG. 7D

526

Instruction Set

000000 00001 00010 00110 00000 100000

FIG. 7E

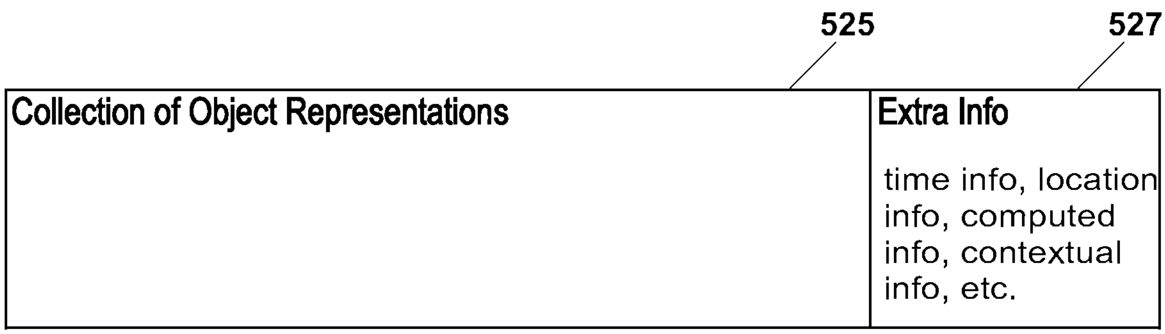

525
527
Collection of Object Representations
Extra Info
time info, location info, computed info, contextual info, etc.

FIG. 8A

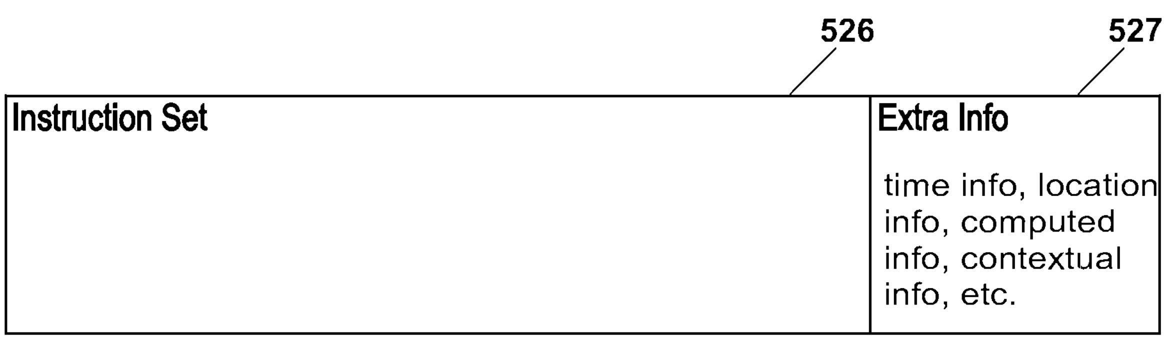

526
527
Instruction Set
Extra Info
time info, location info, computed info, contextual info, etc.

FIG. 8B

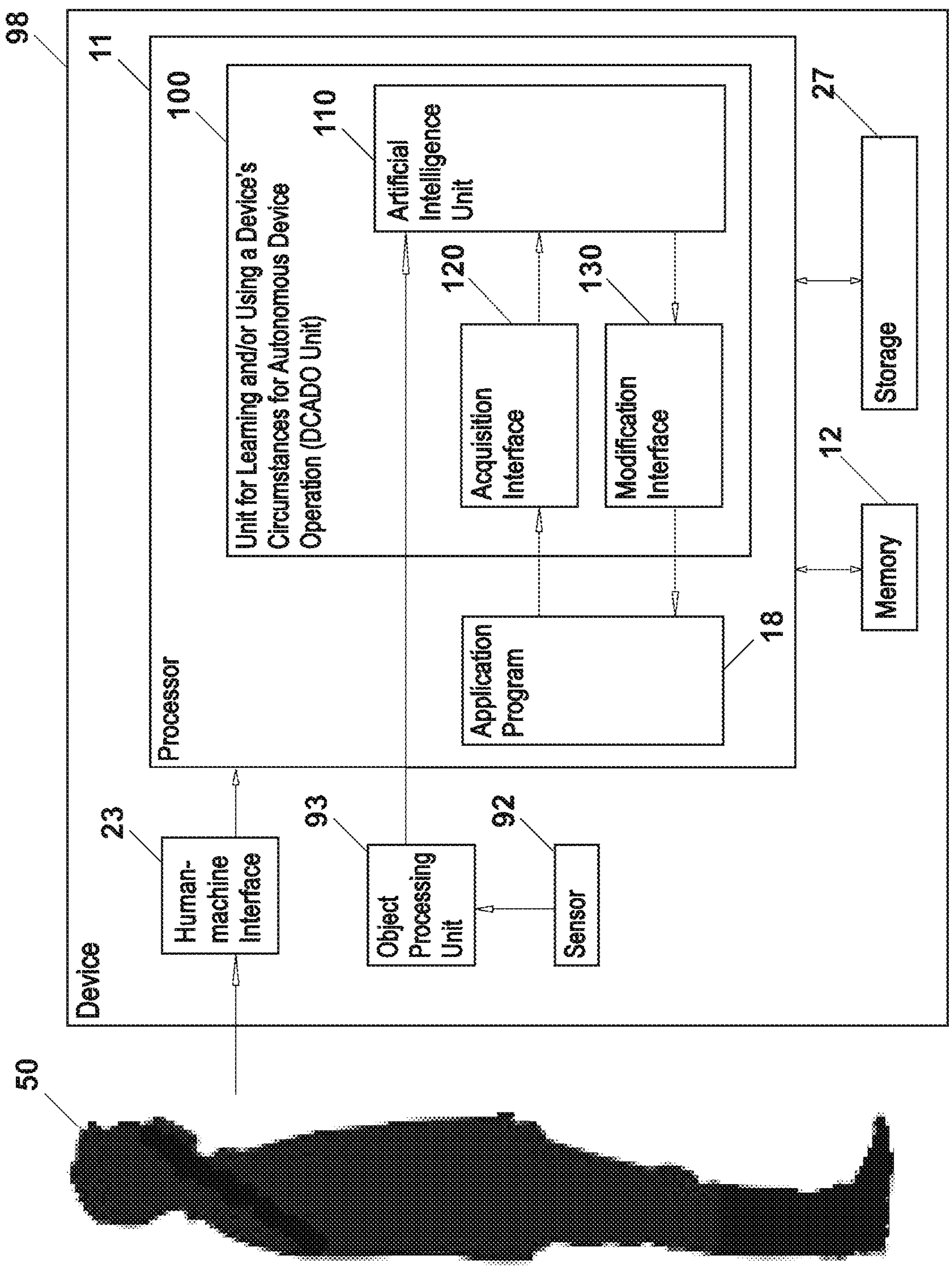

98
Device
11
Processor
100
Unit for Learning and/or Using a Device's Circumstances for Autonomous Device Operation (DCADO Unit)
110
Artificial Intelligence Unit
120
Acquisition Interface
130
Modification Interface
18
Application Program
27
Storage
12
Memory
23
Human-machine Interface
93
Object Processing Unit
92
Sensor
50

FIG. 9

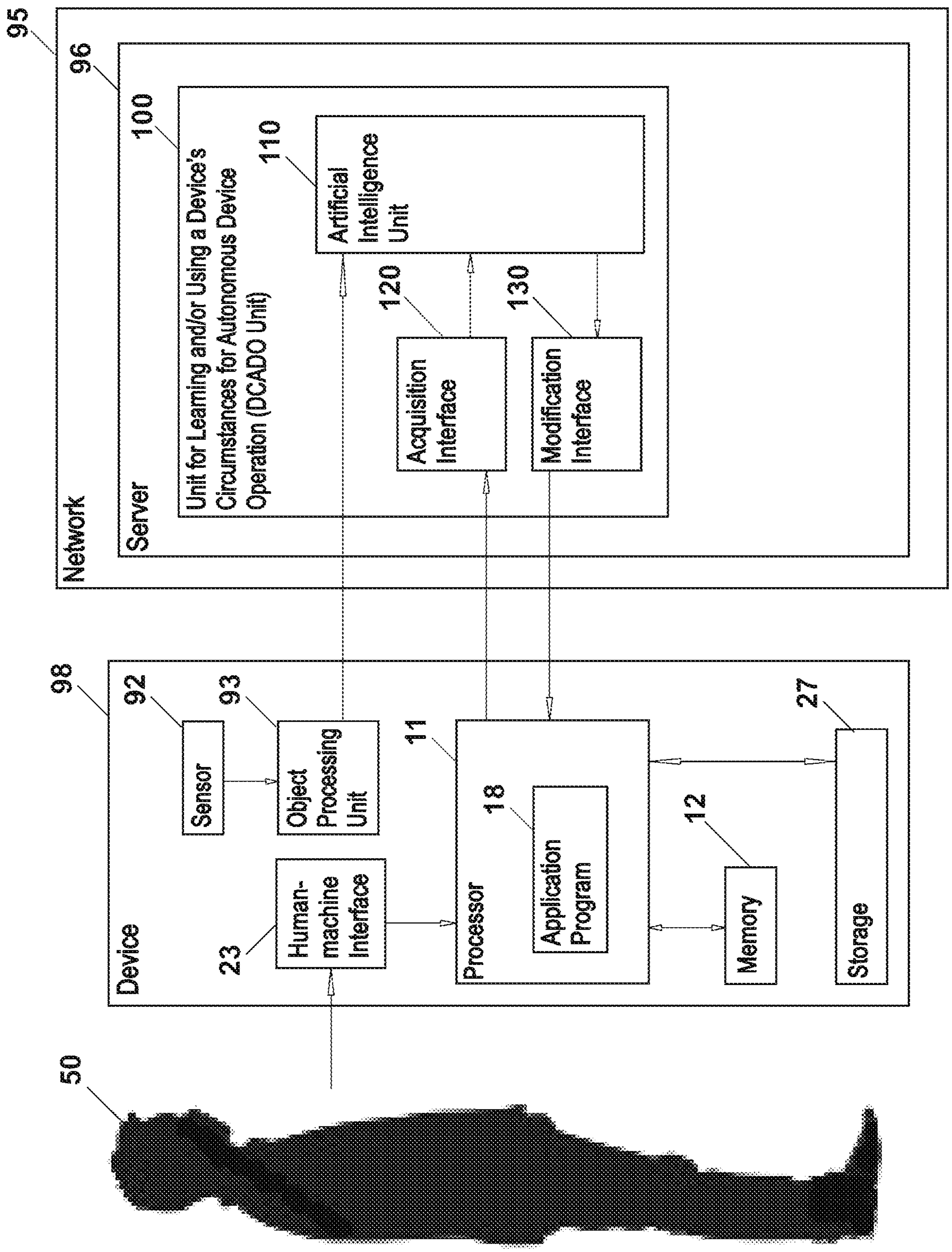

95
96
100
110
Network
Server
Unit for Learning and/or Using a Device's Circumstances for Autonomous Device Operation (DCADO Unit)
Artificial Intelligence Unit
120
130
Acquisition Interface
Modification Interface
98
92
93
11
18
12
27
Device
Sensor
Object Processing Unit
Human-machine Interface
23
Processor
Application Program
Memory
Storage
50

FIG. 10

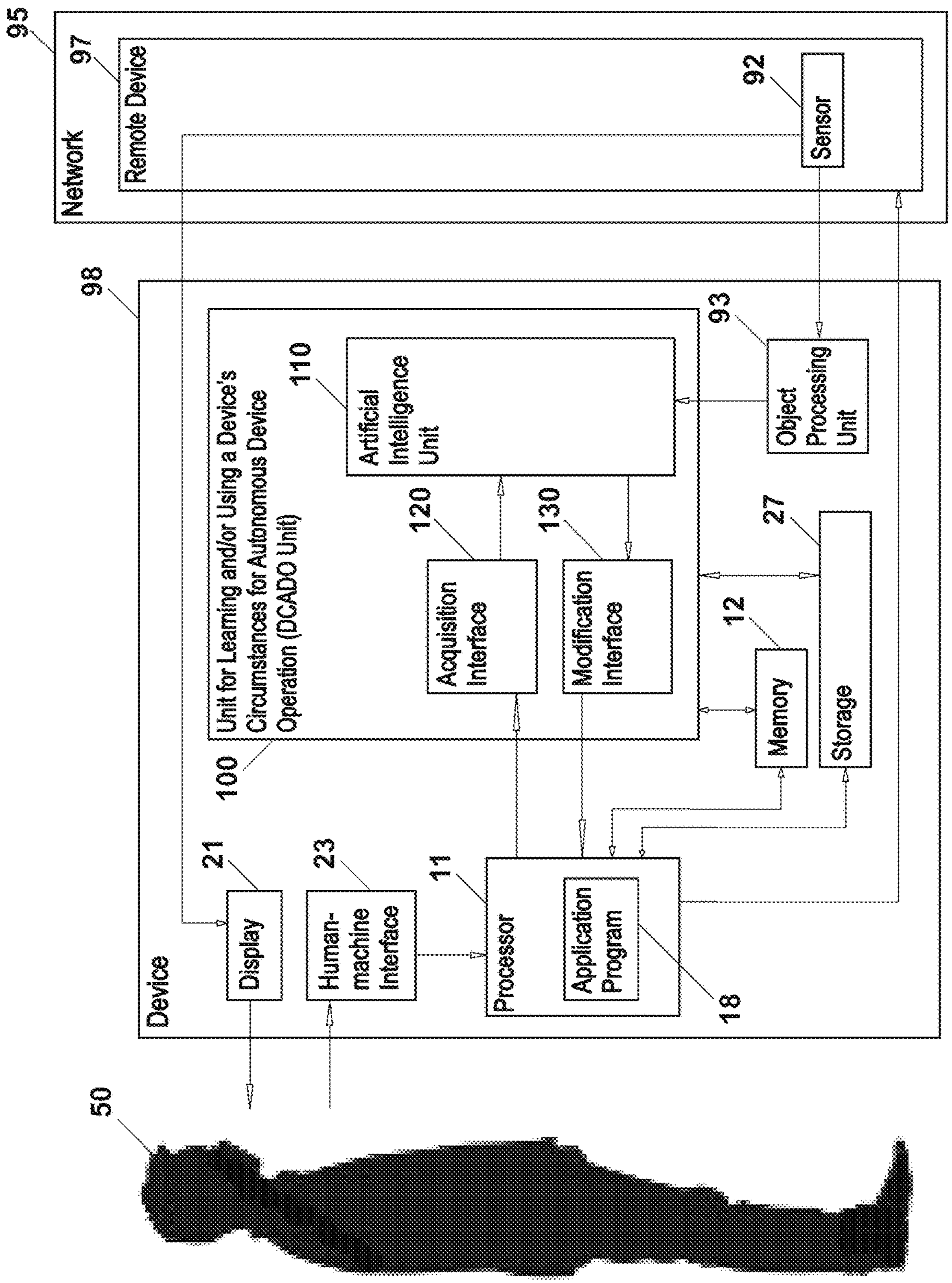

95
Network
97
Remote Device
92
Sensor
98
Device
Unit for Learning and/or Using a Device's Circumstances for Autonomous Device Operation (DCADO Unit)
100
110
Artificial Intelligence Unit
93
Object Processing Unit
120
Acquisition Interface
130
Modification Interface
27
Storage
12
Memory
21
Display
23
Human-machine Interface
11
Processor
Application Program
18
50

FIG. 11

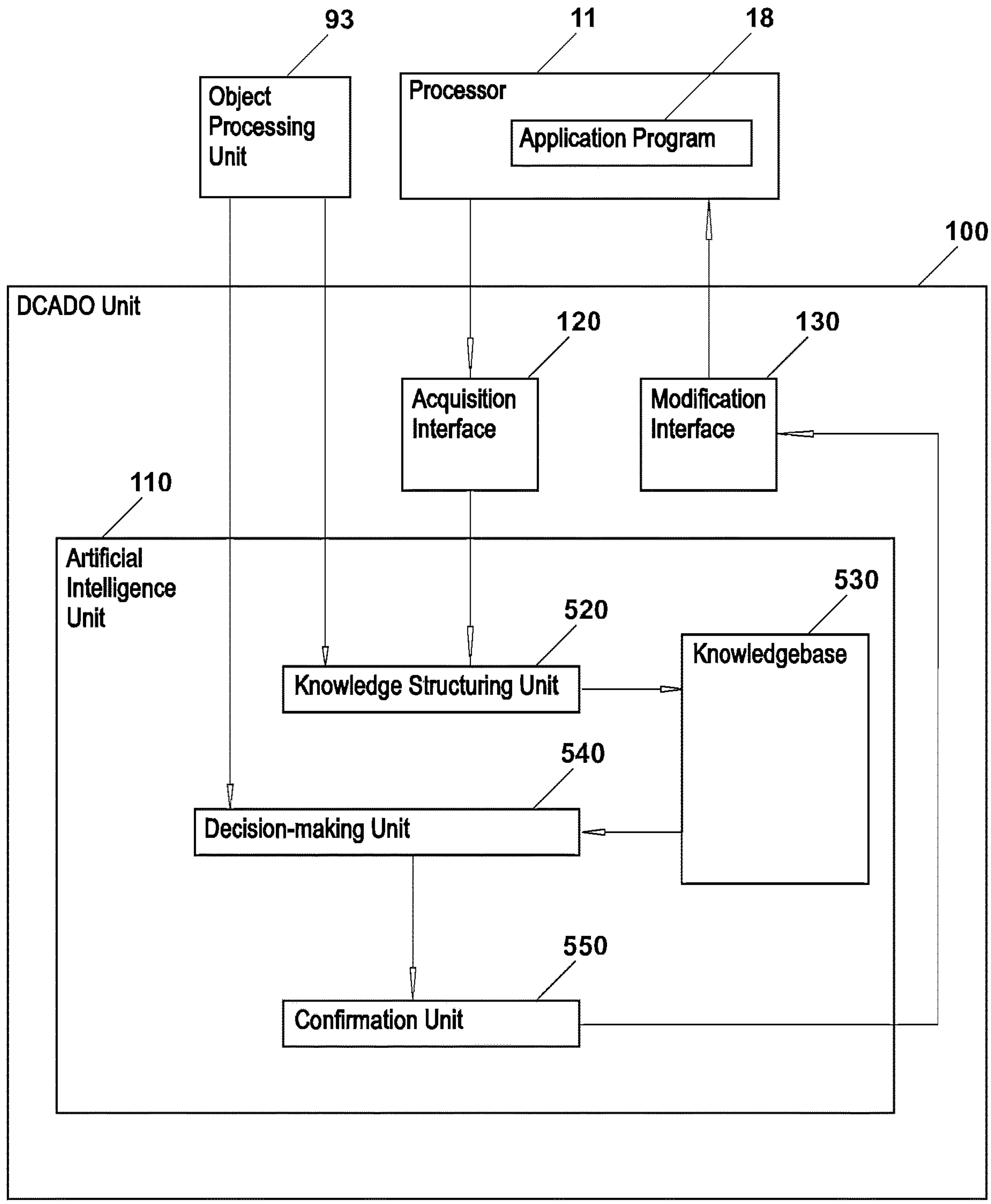
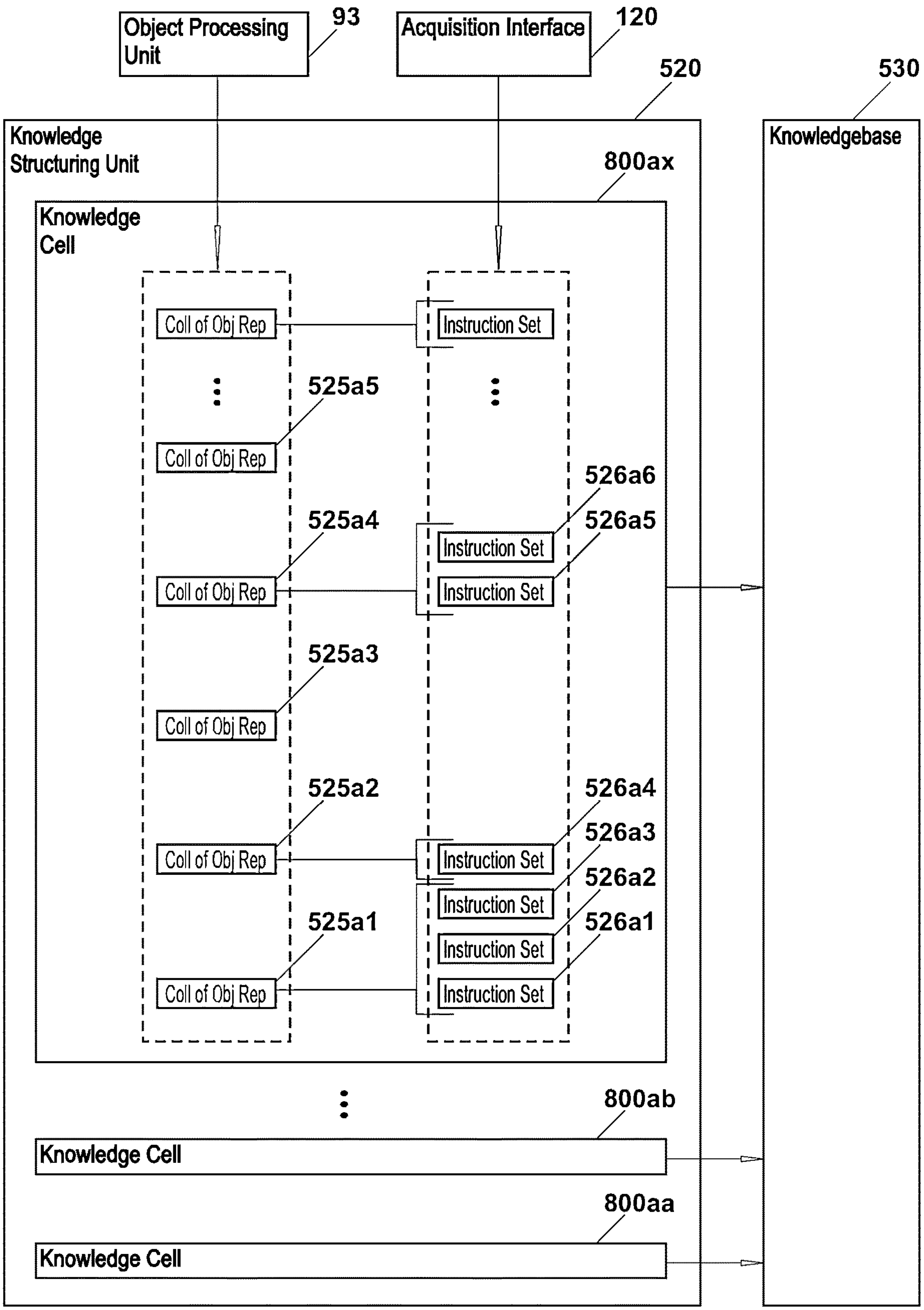
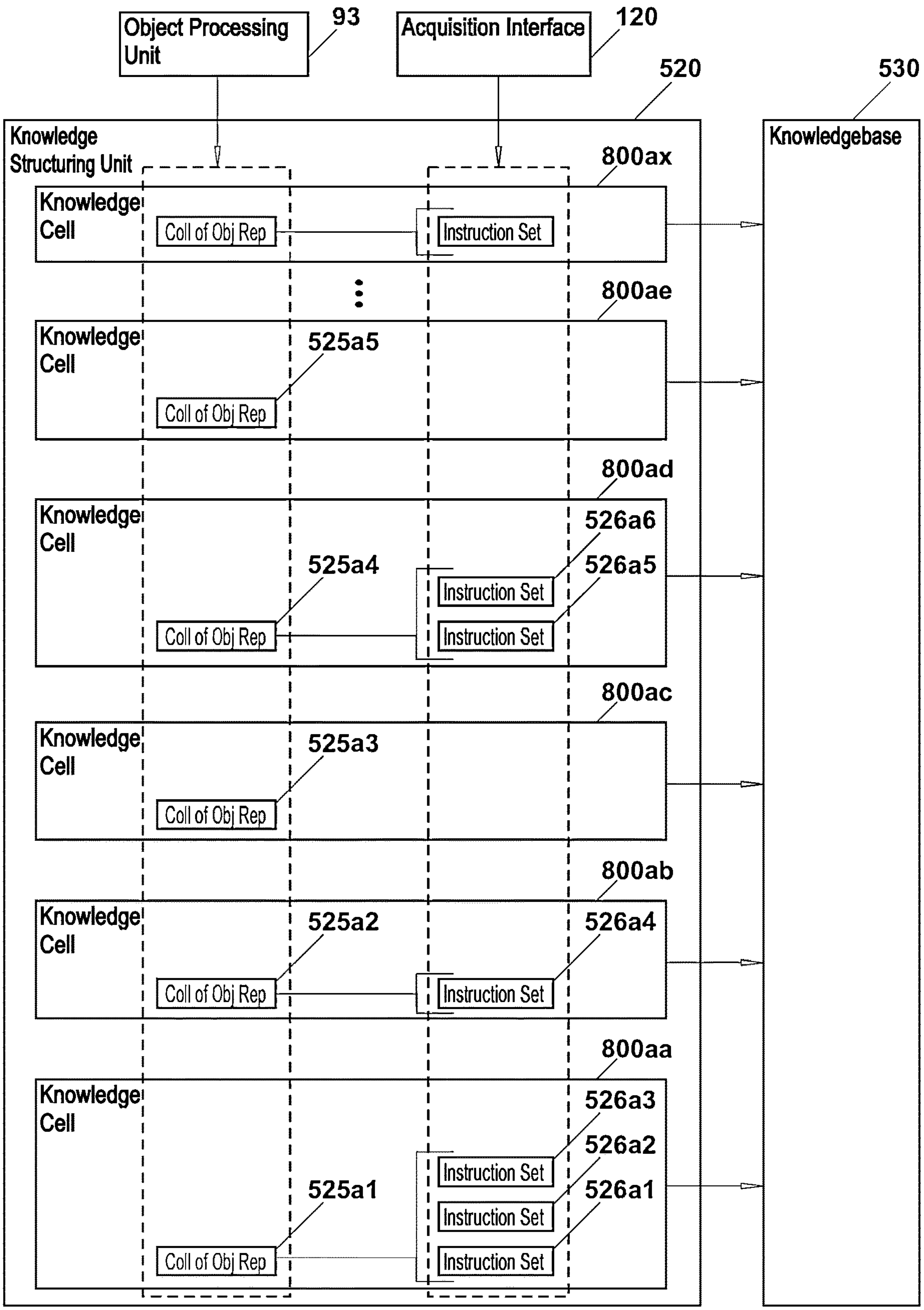

93
11
18
Object Processing Unit
Processor
Application Program
100
DCADO Unit
120
130
Acquisition Interface
Modification Interface
110
Artificial Intelligence Unit
530
520
Knowledge Structuring Unit
Knowledgebase
540
Decision-making Unit
550
Confirmation Unit

FIG. 12



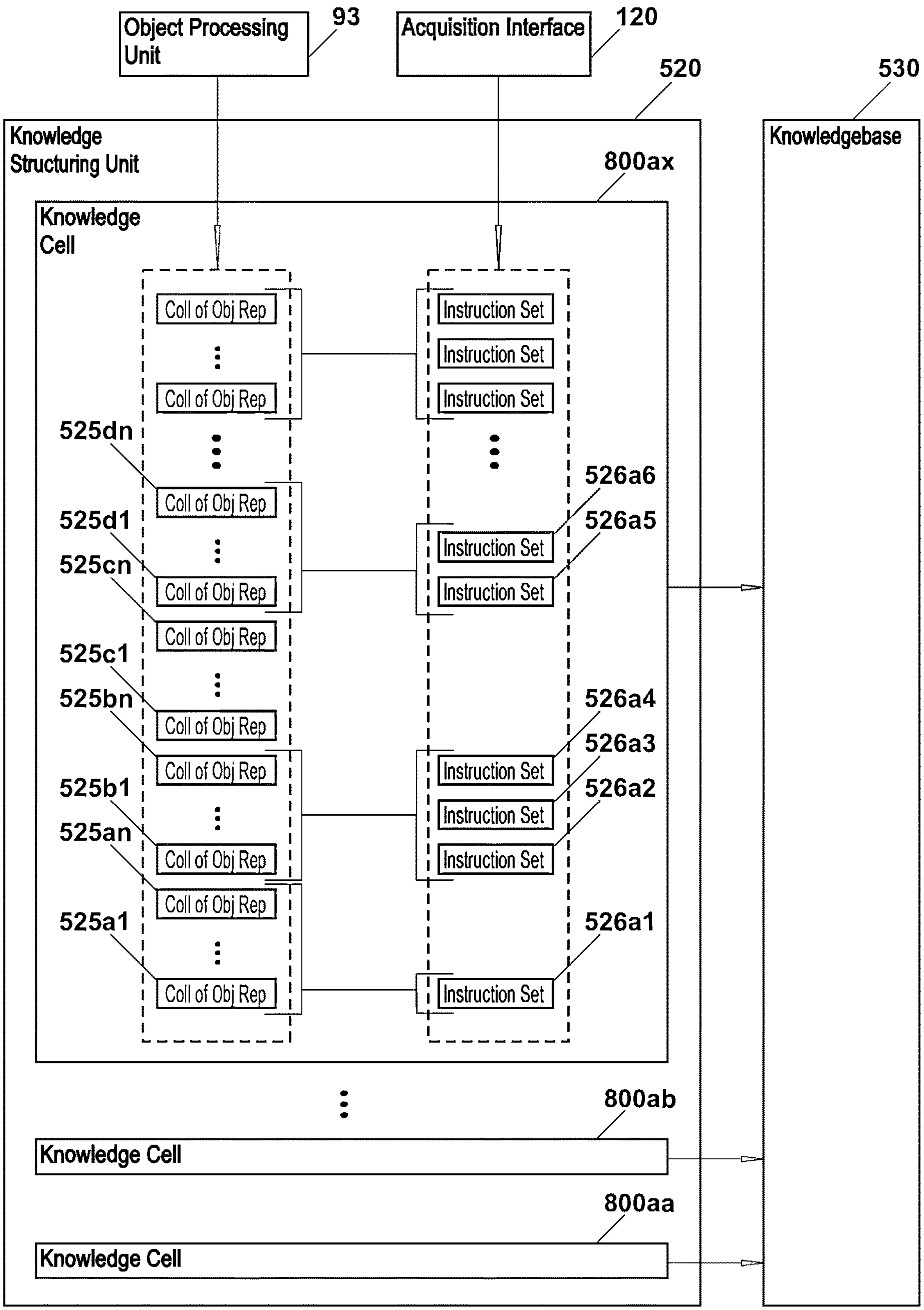
Object Processing Unit
93
Acquisition Interface
120
520
530
Knowledge Structuring Unit
800ax
Knowledgebase
Knowledge Cell
Coll of Obj Rep
Instruction Set
525a5
Coll of Obj Rep
526a6
525a4
526a5
Instruction Set
Coll of Obj Rep
Instruction Set
525a3
Coll of Obj Rep
525a2
526a4
526a3
Coll of Obj Rep
Instruction Set
526a2
Instruction Set
525a1
526a1
Instruction Set
Coll of Obj Rep
Instruction Set
800ab
Knowledge Cell
800aa
Knowledge Cell

FIG. 13



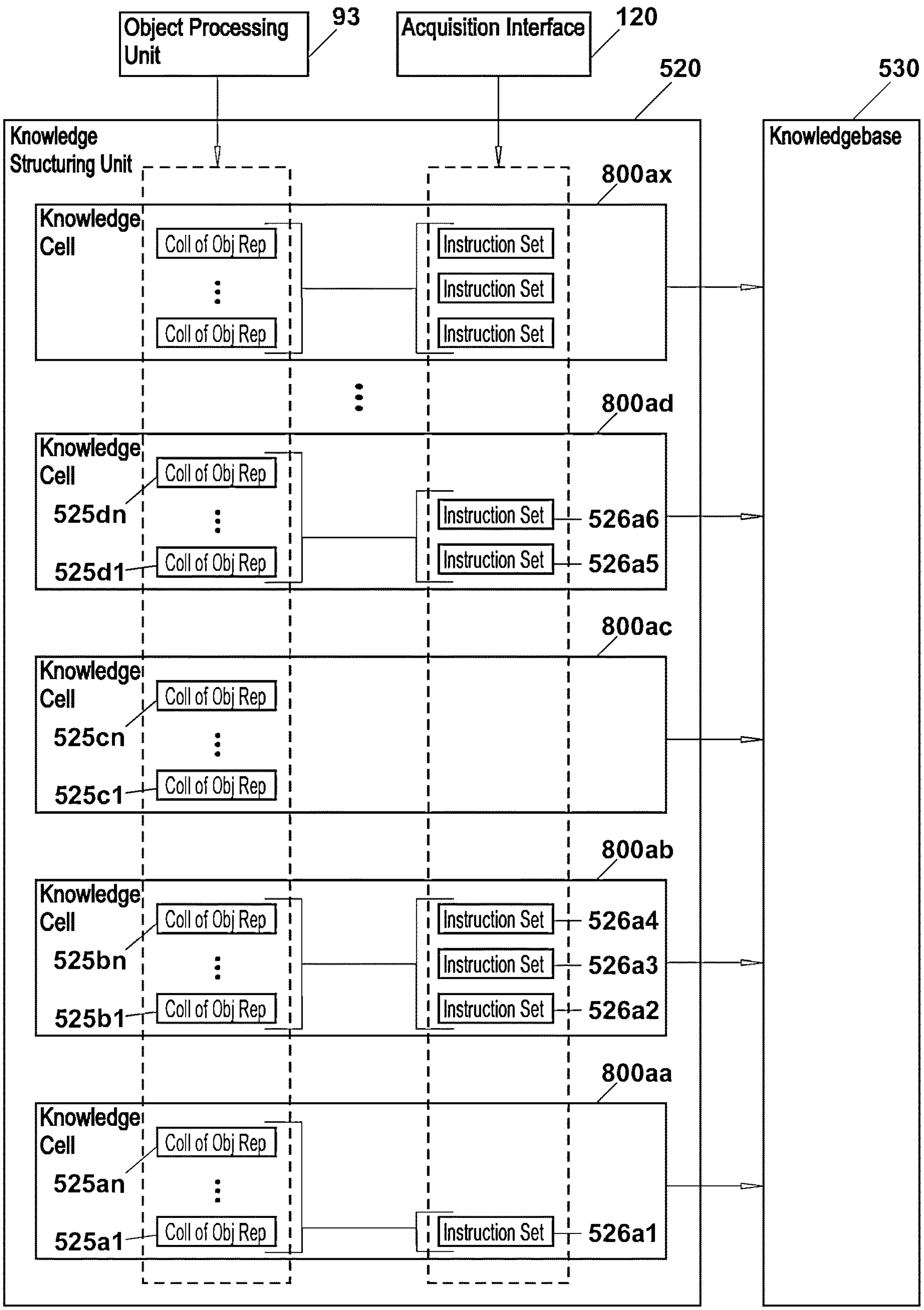
Object Processing Unit
93
Acquisition Interface
120
520
530
Knowledge Structuring Unit
Knowledgebase
800ax
Knowledge Cell
Coll of Obj Rep
Instruction Set
800ae
Knowledge Cell
525a5
Coll of Obj Rep
800ad
Knowledge Cell
526a6
525a4
526a5
Instruction Set
Coll of Obj Rep
Instruction Set
800ac
Knowledge Cell
525a3
Coll of Obj Rep
800ab
Knowledge Cell
525a2
526a4
Coll of Obj Rep
Instruction Set
800aa
Knowledge Cell
526a3
526a2
525a1
526a1
Instruction Set
Instruction Set
Coll of Obj Rep
Instruction Set

FIG. 14



Object Processing Unit
93
Acquisition Interface
120
520
530
Knowledge Structuring Unit
Knowledgebase
800ax
Knowledge Cell
Coll of Obj Rep
Instruction Set
525dn
526a6
525d1
526a5
525cn
525c1
525bn
526a4
526a3
525b1
526a2
525an
525a1
526a1
800ab
Knowledge Cell
800aa
Knowledge Cell

FIG. 15



Object Processing Unit
93
Acquisition Interface
120
520
530
Knowledge Structuring Unit
Knowledgebase
800ax
Knowledge Cell
Coll of Obj Rep
Coll of Obj Rep
Instruction Set
Instruction Set
Instruction Set
800ad
Knowledge Cell
525dn
525d1
Coll of Obj Rep
Coll of Obj Rep
Instruction Set
Instruction Set
526a6
526a5
800ac
Knowledge Cell
525cn
525c1
Coll of Obj Rep
Coll of Obj Rep
800ab
Knowledge Cell
525bn
525b1
Coll of Obj Rep
Coll of Obj Rep
Instruction Set
Instruction Set
Instruction Set
526a4
526a3
526a2
800aa
Knowledge Cell
525an
525a1
Coll of Obj Rep
Coll of Obj Rep
Instruction Set
526a1

FIG. 16

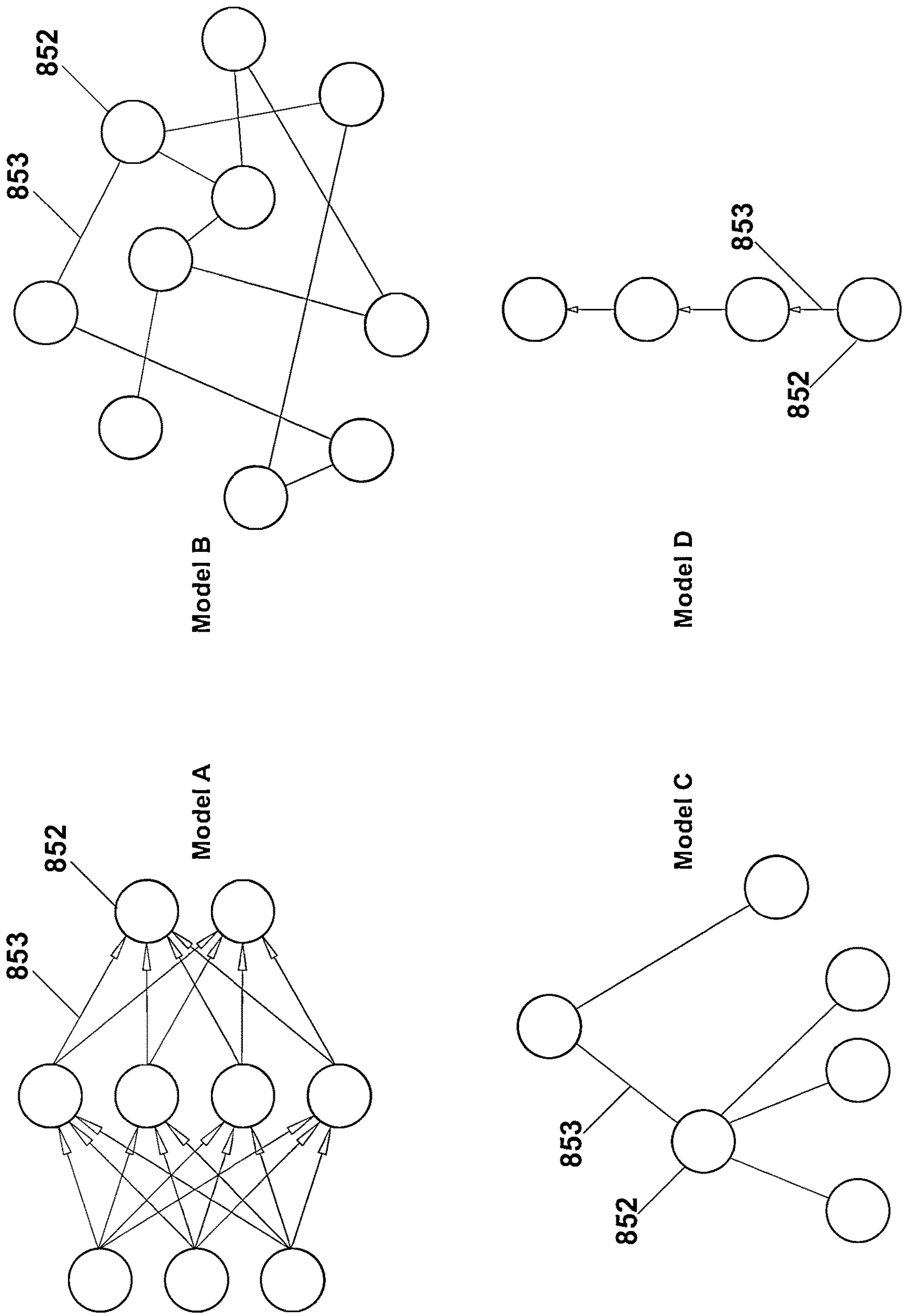


FIG. 17

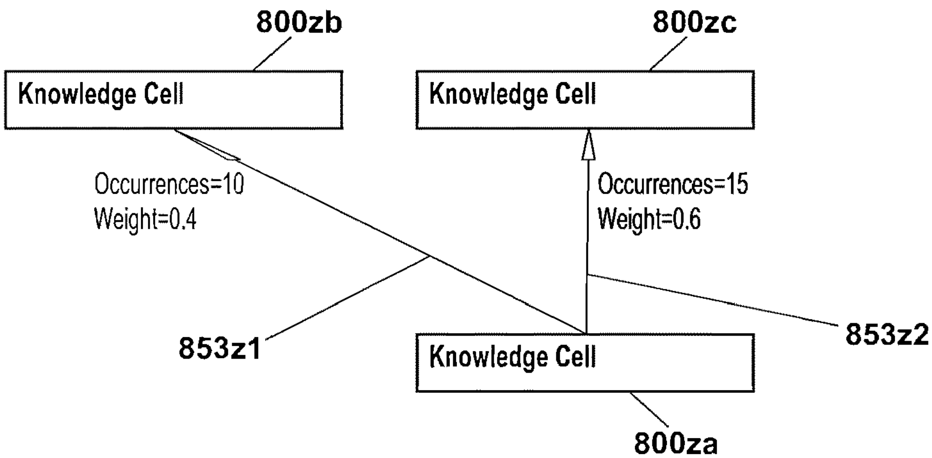


FIG. 18A

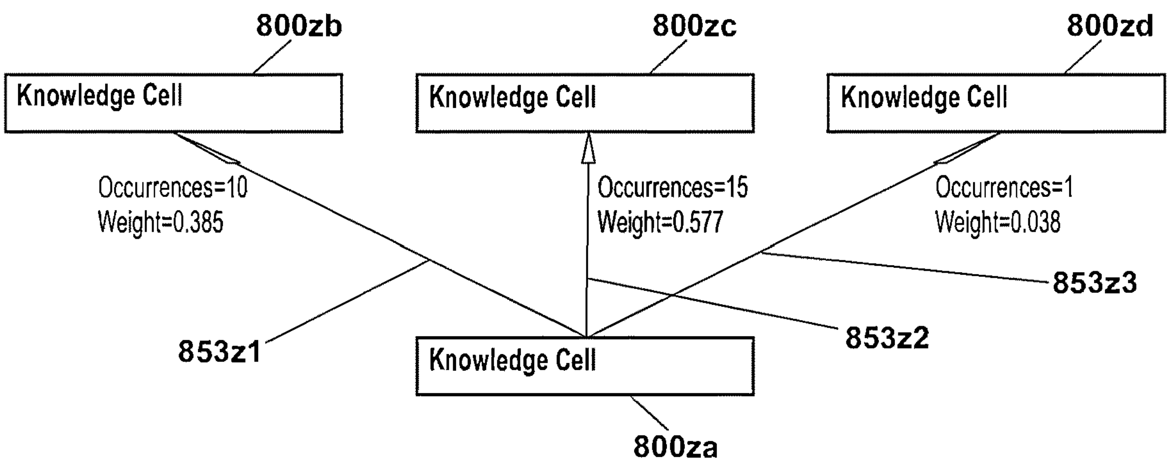


FIG. 18B

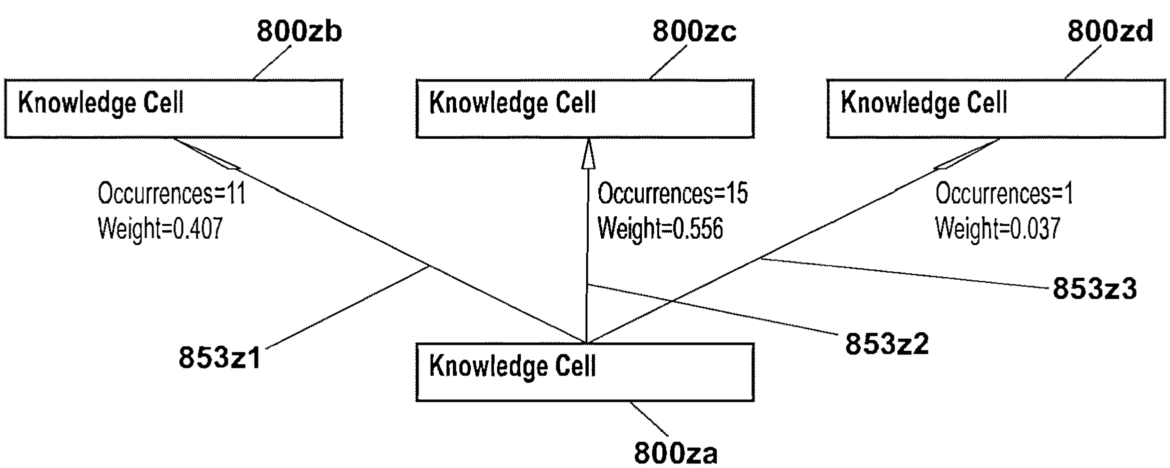


FIG. 18C

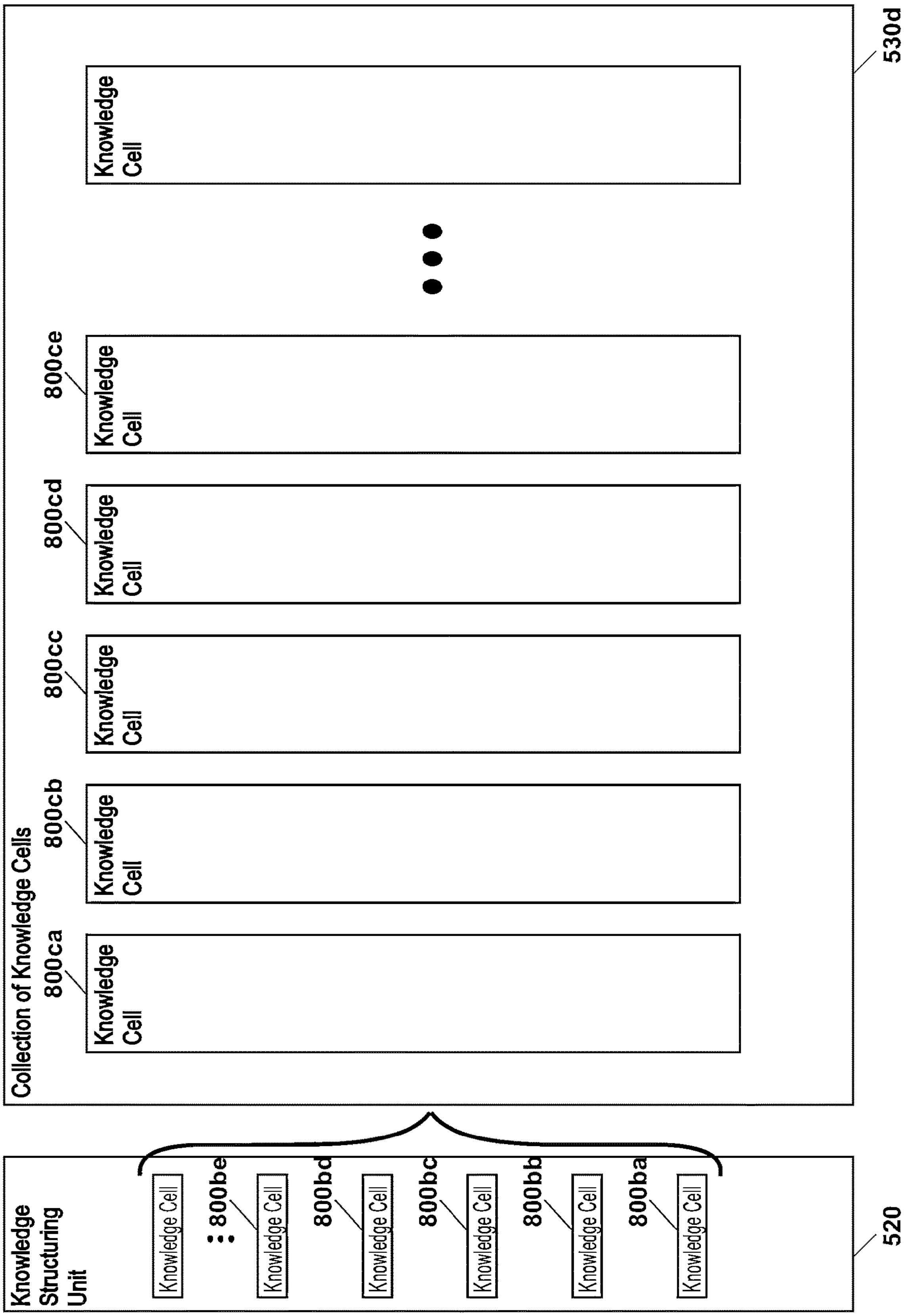

Knowledge Structuring Unit
Knowledge Cell
800ba
800bb
800bc
800bd
800be
520
Collection of Knowledge Cells
Knowledge Cell
800ca
800cb
800cc
800cd
800ce
530d

FIG. 19

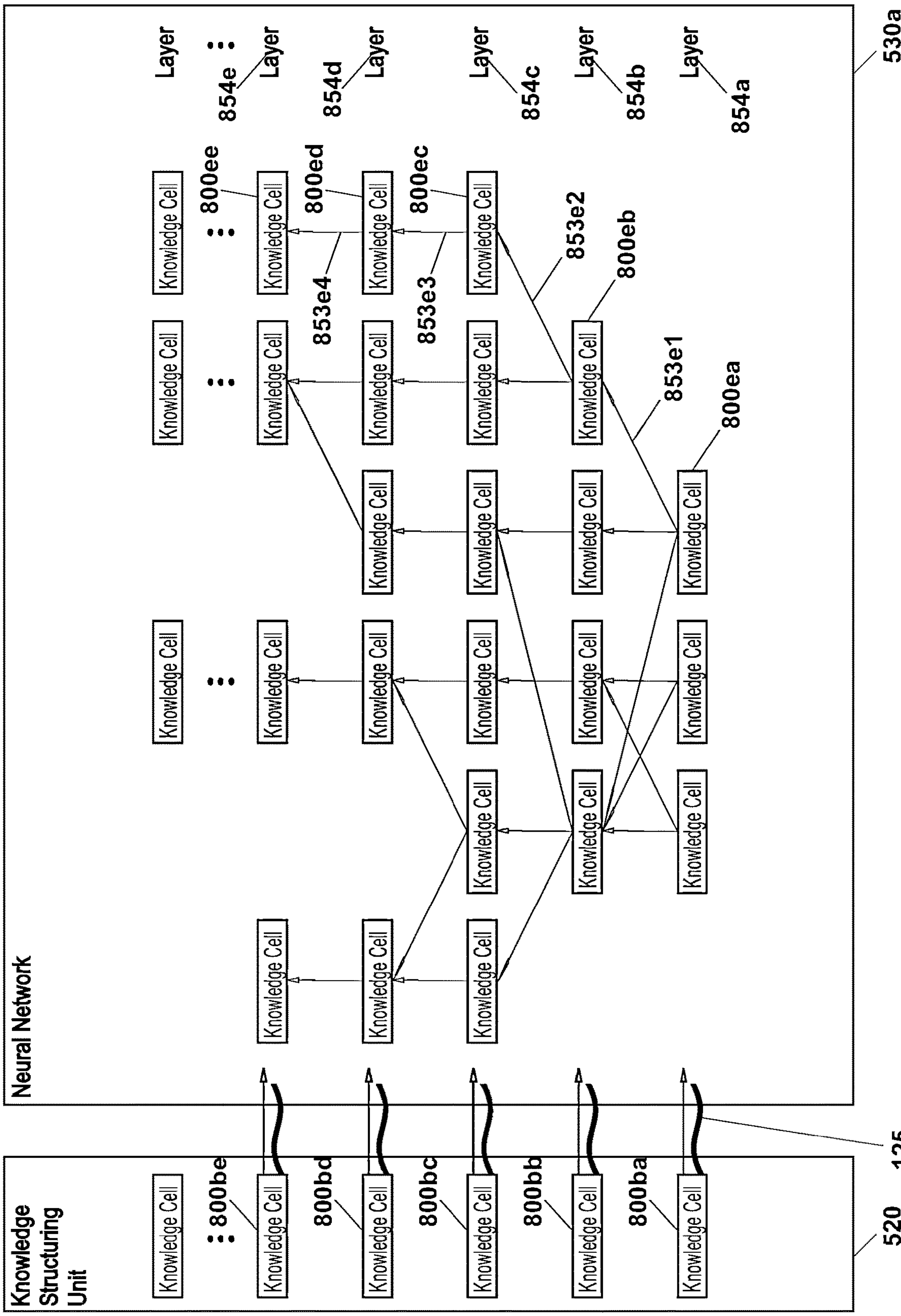

Neural Network
Knowledge Structuring Unit
Knowledge Cell
Layer
854a
854b
854c
854d
854e
800ea
800eb
800ec
800ed
800ee
853e1
853e2
853e3
853e4
800ba
800bb
800bc
800bd
800be
530a
520
125

FIG. 20

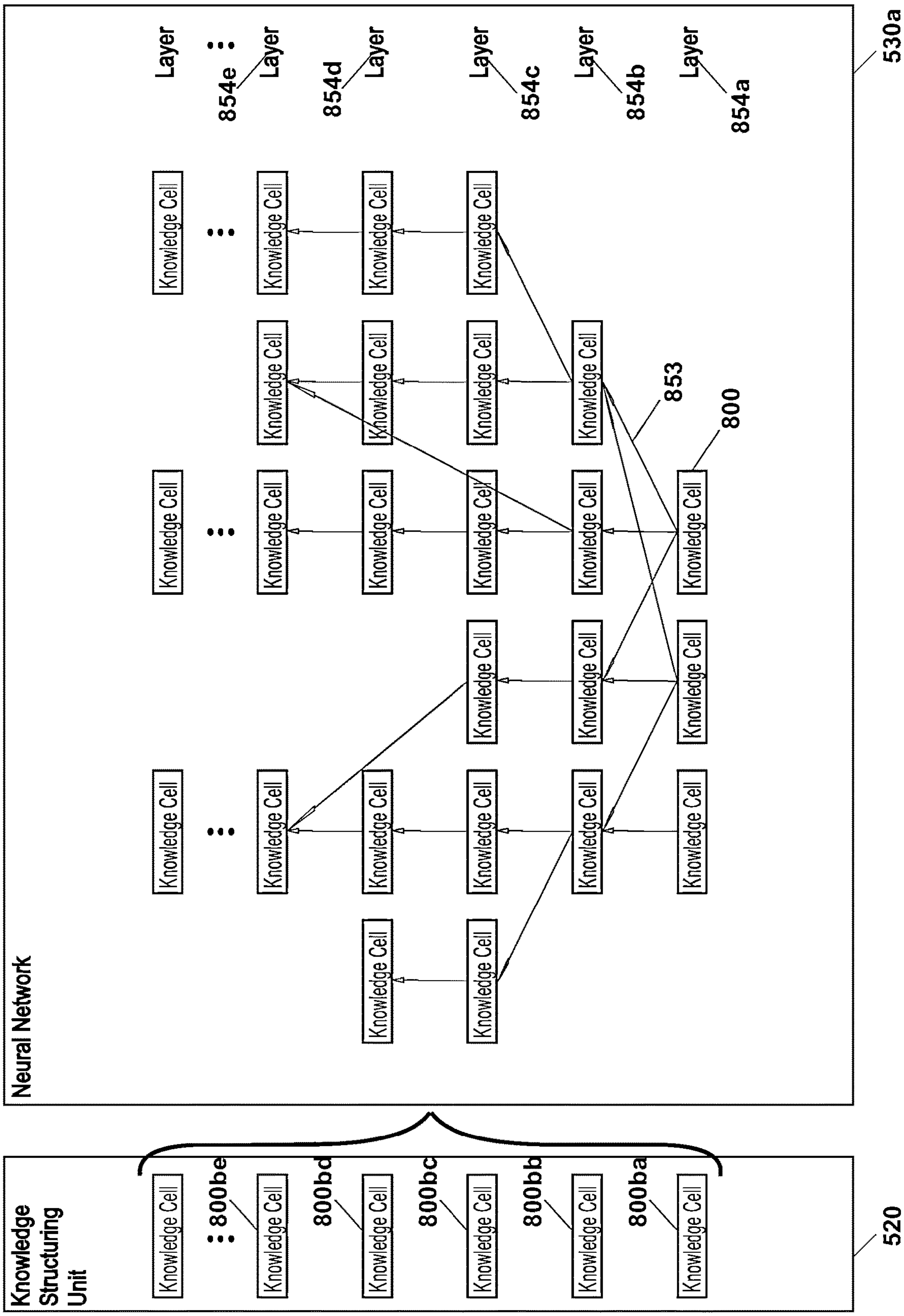

Knowledge Structuring Unit
520
Knowledge Cell
800ba
800bb
800bc
800bd
800be
Neural Network
530a
Layer
854a
854b
854c
854d
854e
853
800

FIG. 21

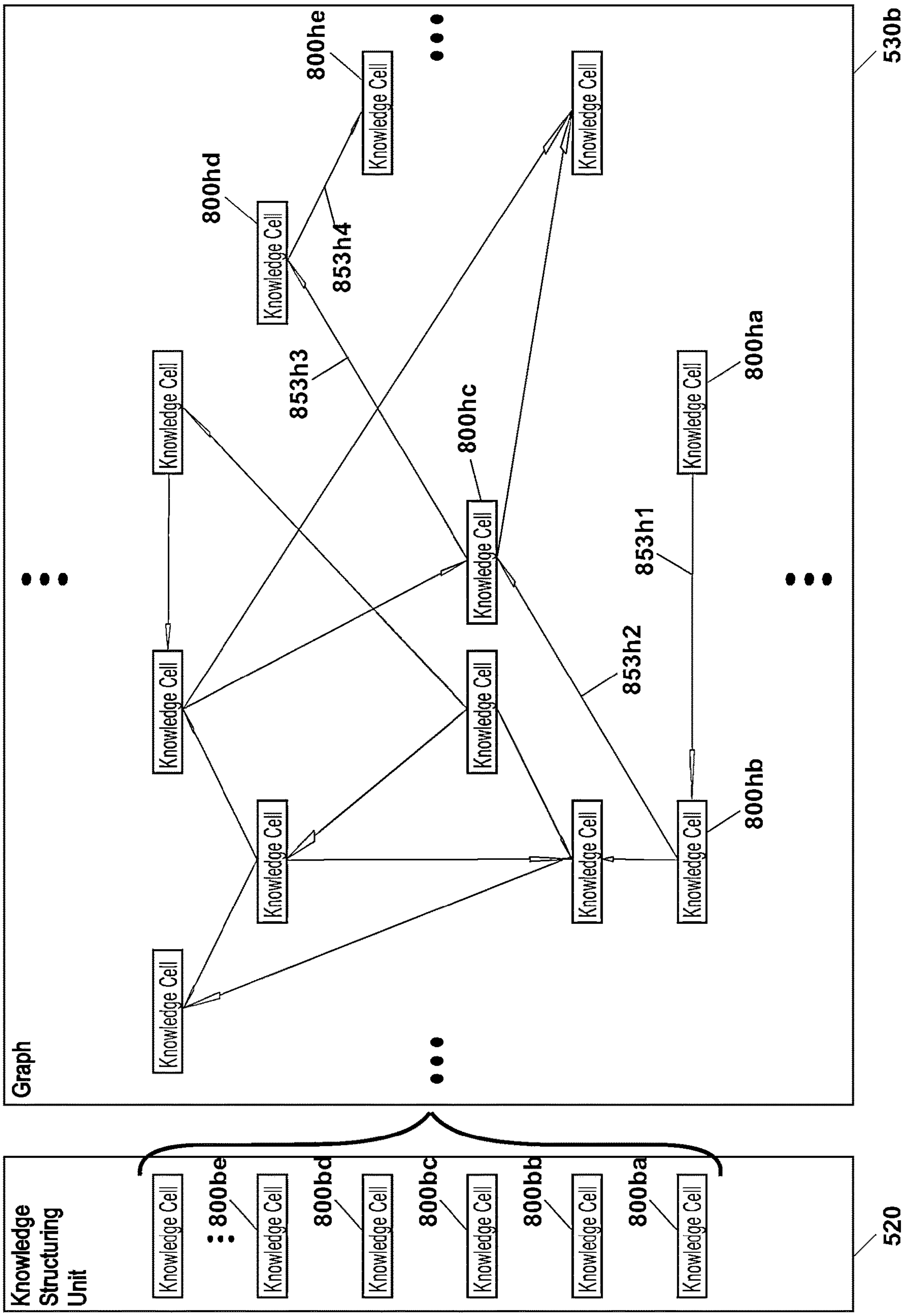

Graph
Knowledge Cell
800ha
800hb
800hc
800hd
800he
853h1
853h2
853h3
853h4
530b
Knowledge Structuring Unit
800ba
800bb
800bc
800bd
800be
520

FIG. 22

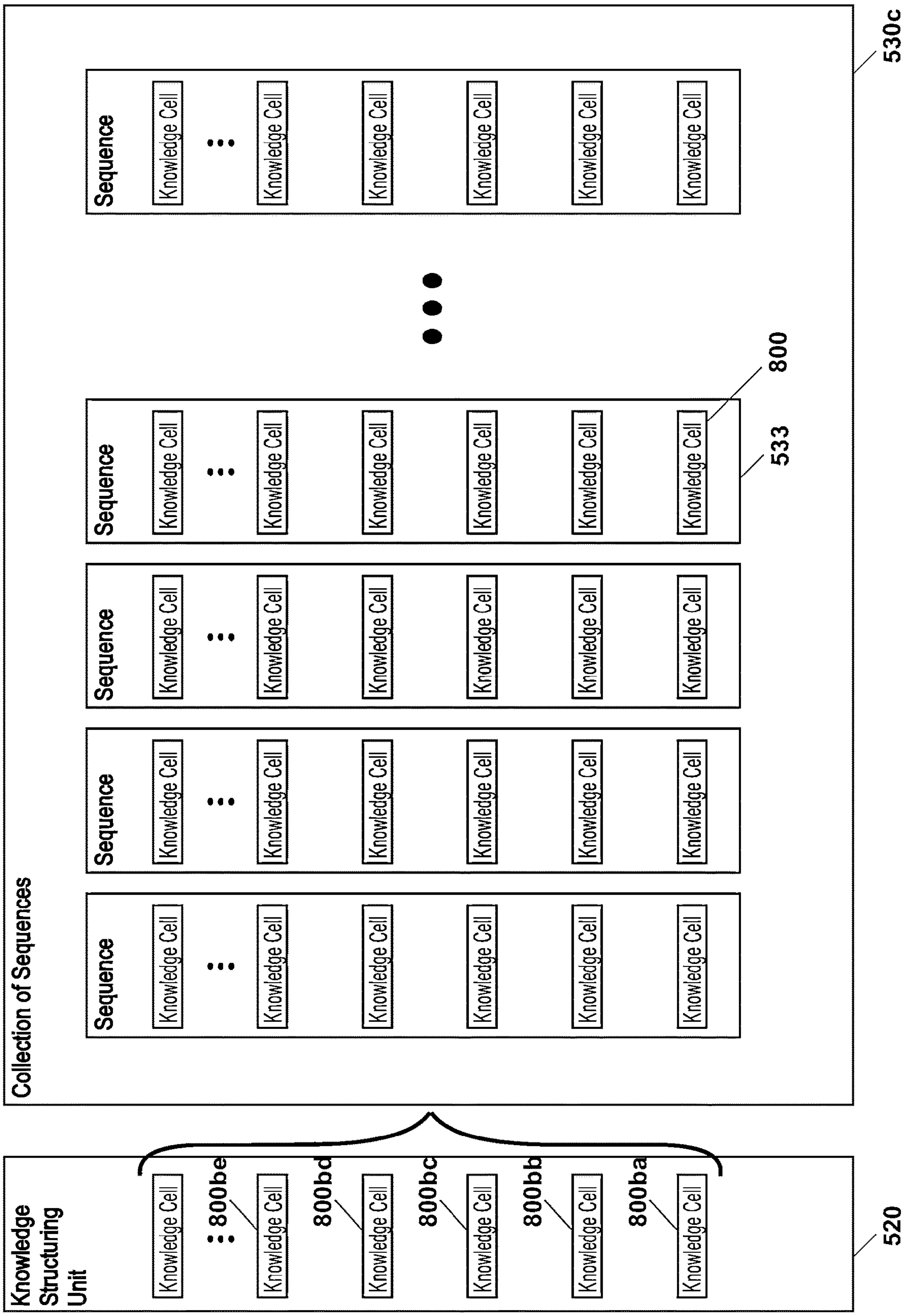

Collection of Sequences
Sequence
Knowledge Cell
530c
533
800
Knowledge Structuring Unit
800be
800bd
800bc
800bb
800ba
520

FIG. 23

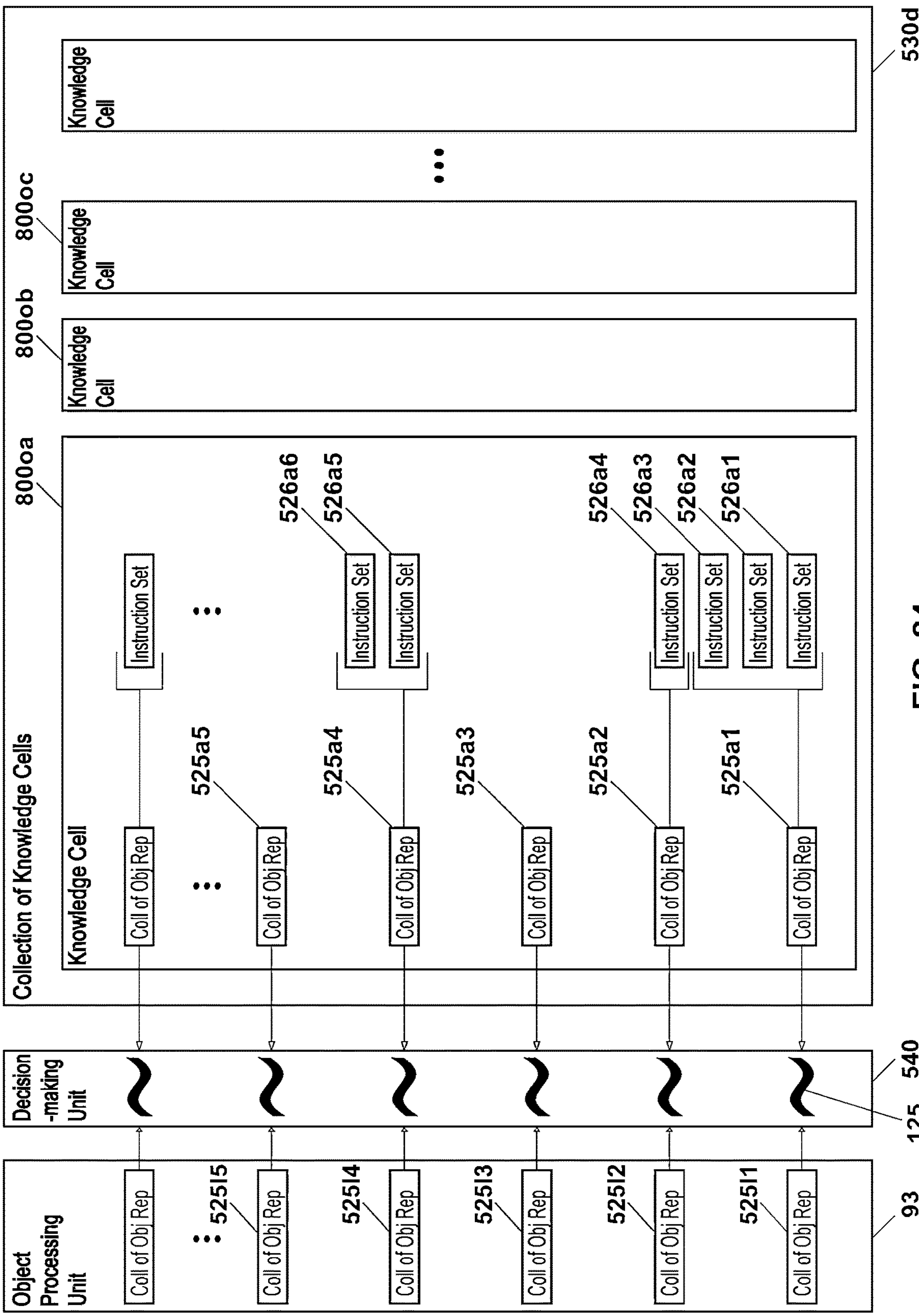

Object Processing Unit
Coll of Obj Rep
525l1
525l2
525l3
525l4
525l5
93
Decision -making Unit
540
125
Collection of Knowledge Cells
Knowledge Cell
Instruction Set
525a1
525a2
525a3
525a4
525a5
526a1
526a2
526a3
526a4
526a5
526a6
800oa
800ob
800oc
530d

FIG. 24

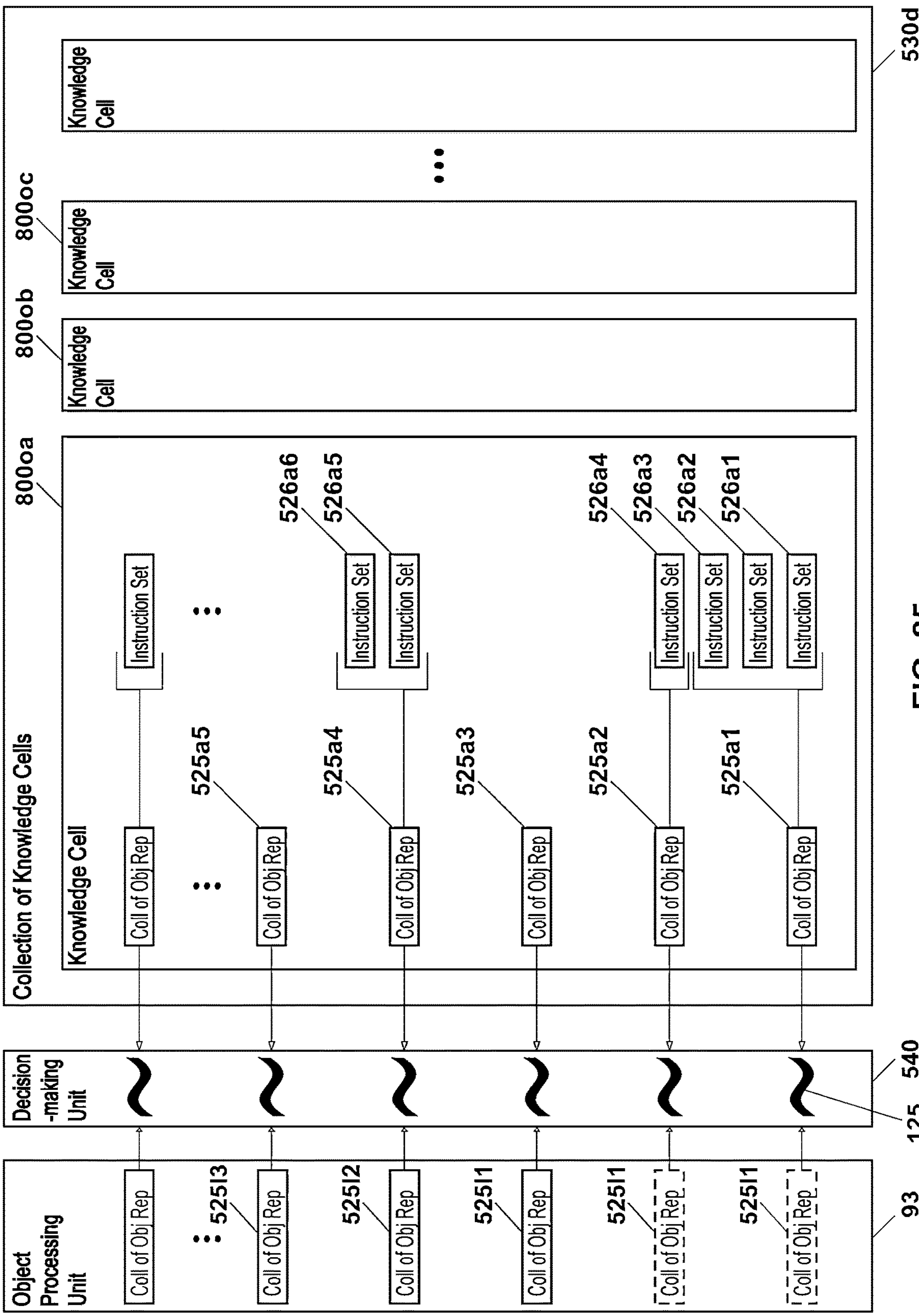

Object Processing Unit
Coll of Obj Rep
525l3
525l2
525l1
525l1
525l1
93
Decision -making Unit
540
125
Collection of Knowledge Cells
Knowledge Cell
800oa
525a5
525a4
525a3
525a2
525a1
Instruction Set
526a6
526a5
526a4
526a3
526a2
526a1
800ob
800oc
Knowledge Cell
530d

FIG. 25

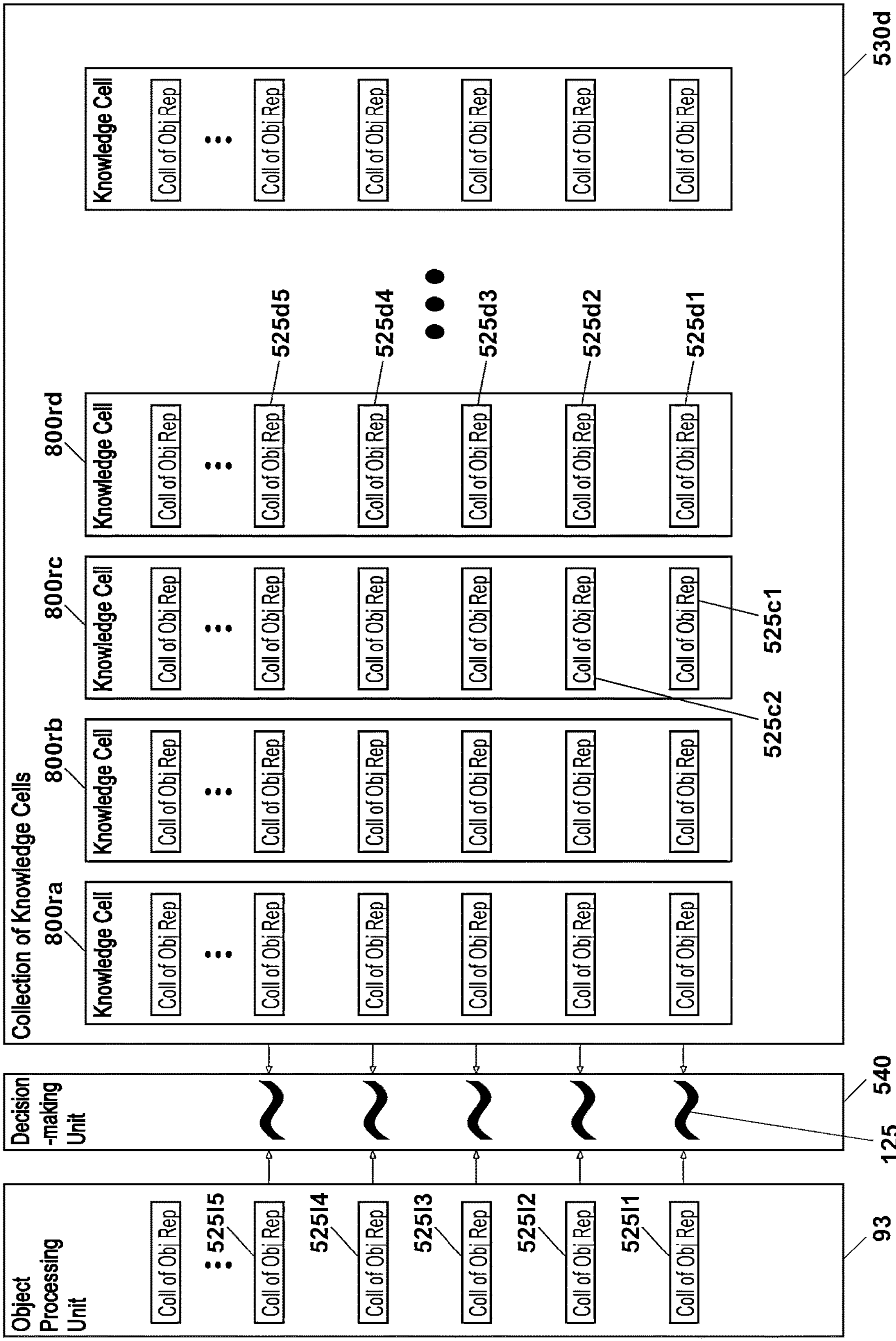

Object Processing Unit
Coll of Obj Rep
525l5
525l4
525l3
525l2
525l1
93
Decision -making Unit
125
540
Collection of Knowledge Cells
Knowledge Cell
800ra
800rb
800rc
800rd
525c1
525c2
525d5
525d4
525d3
525d2
525d1
530d

FIG. 26

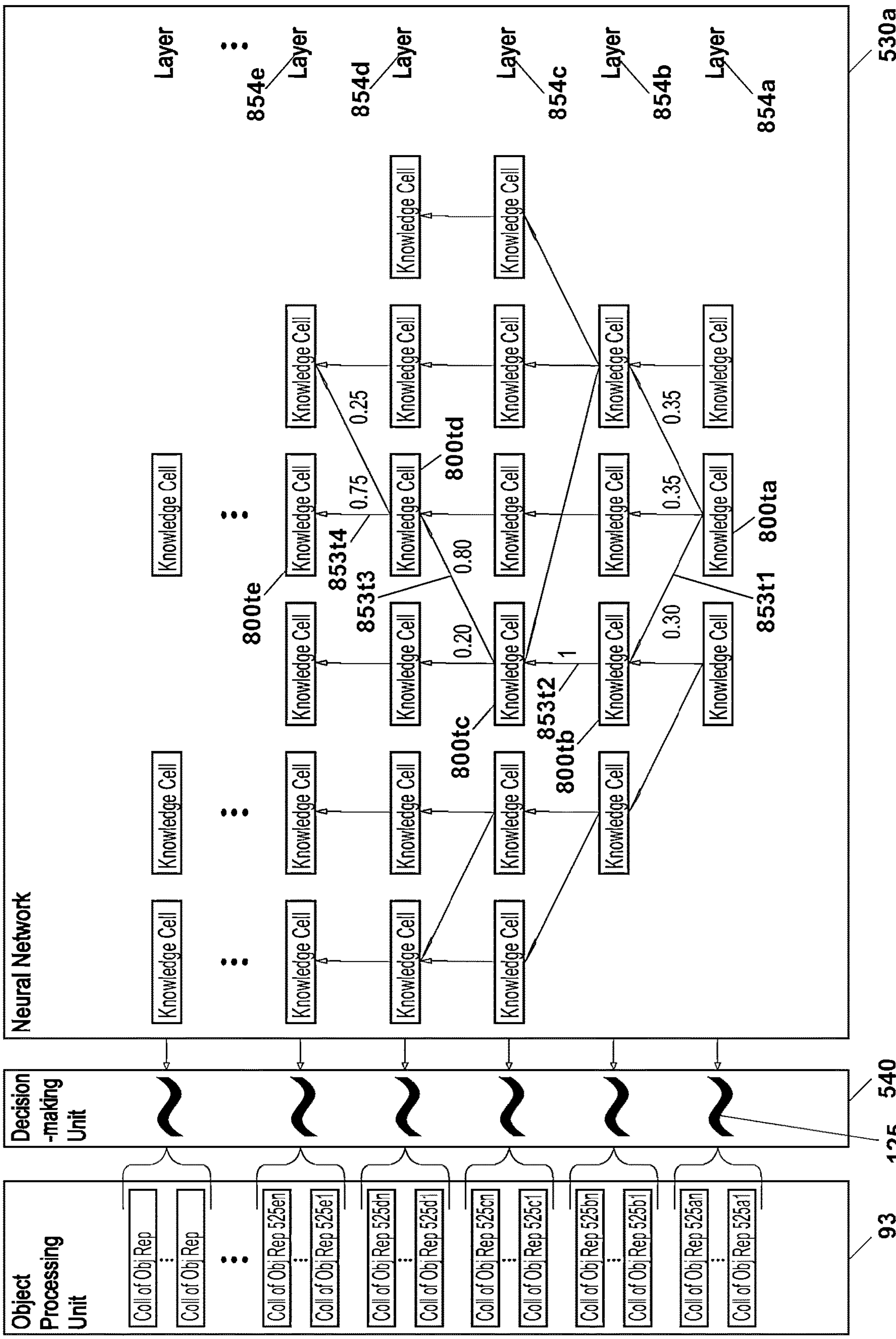

Neural Network
Decision -making Unit
Object Processing Unit
Knowledge Cell
Layer
854a
854b
854c
854d
854e
800ta
800tb
800tc
800td
800te
853t1
853t2
853t3
853t4
0.25
0.75
0.35
0.35
0.80
0.20
0.30
1
530a
540
125
93
Coll of Obj Rep 525an
Coll of Obj Rep 525a1
Coll of Obj Rep 525bn
Coll of Obj Rep 525b1
Coll of Obj Rep 525cn
Coll of Obj Rep 525c1
Coll of Obj Rep 525dn
Coll of Obj Rep 525d1
Coll of Obj Rep 525en
Coll of Obj Rep 525e1
Coll of Obj Rep

FIG. 27

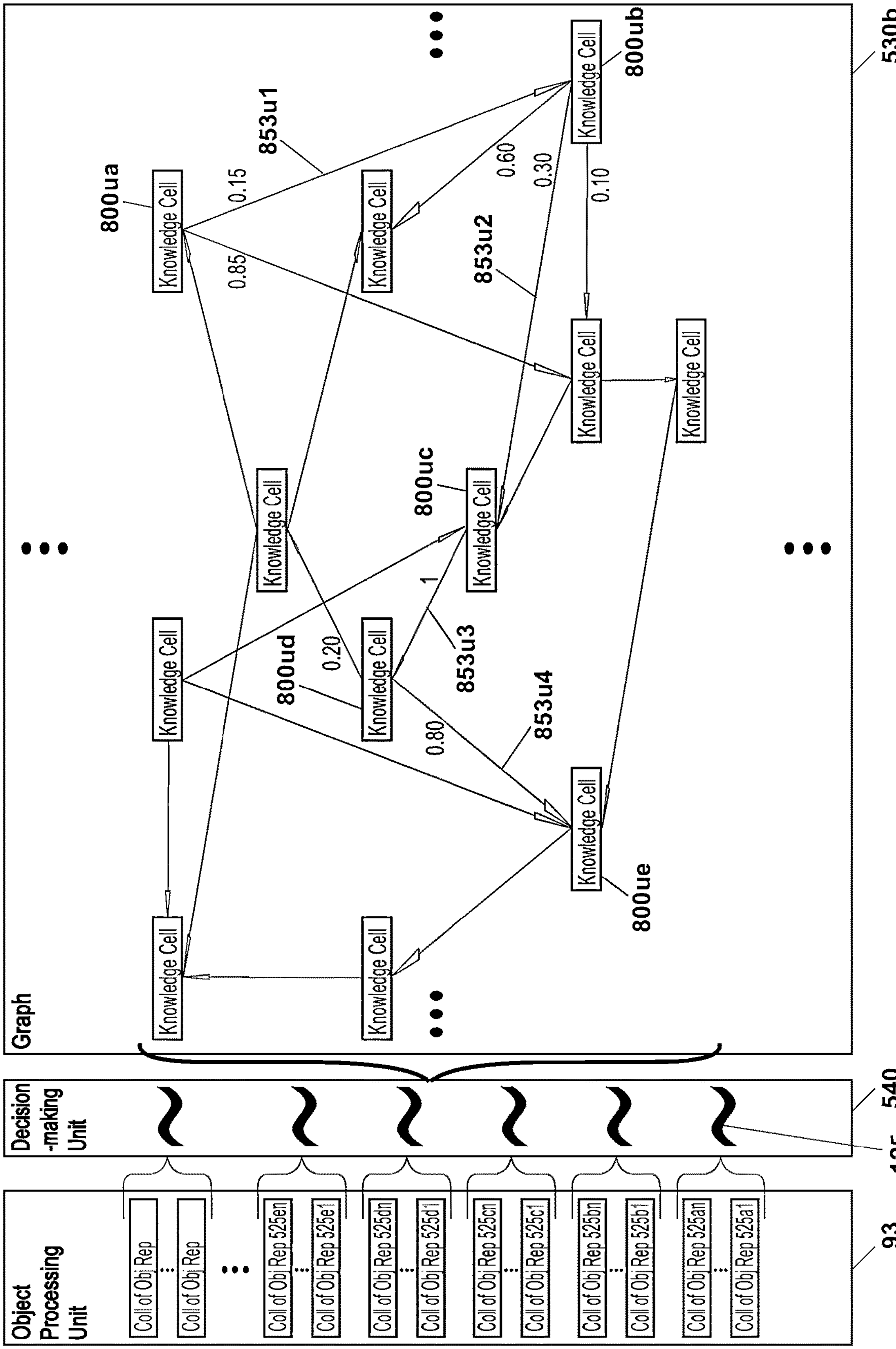
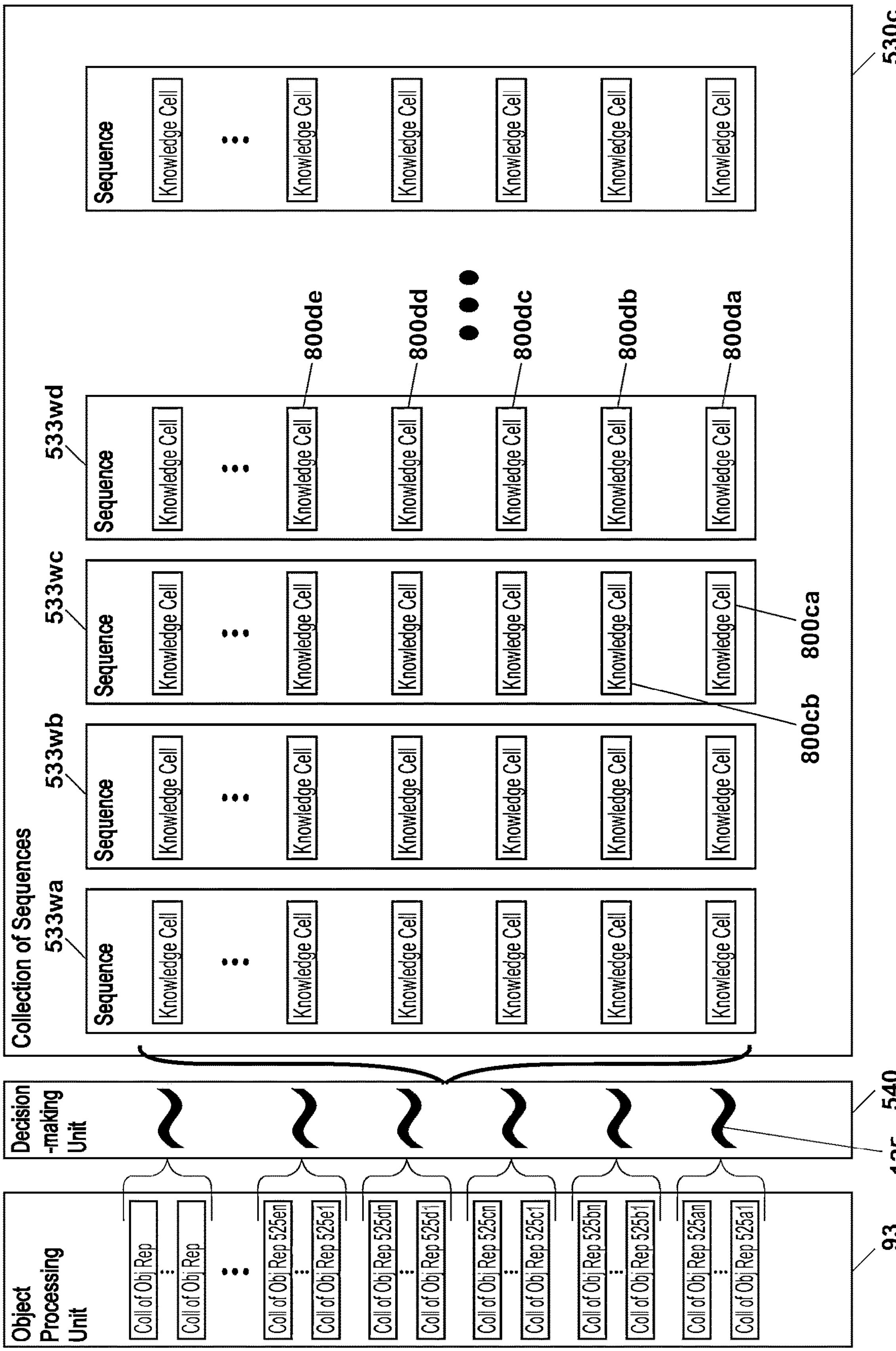

Graph
Knowledge Cell
800ua
800ub
800uc
800ud
800ue
853u1
853u2
853u3
853u4
0.15
0.85
0.60
0.30
0.10
0.20
0.80
1
530b
Decision
-making
Unit
540
125
Object
Processing
Unit
Coll of Obj Rep
Coll of Obj Rep 525en
Coll of Obj Rep 525e1
Coll of Obj Rep 525dn
Coll of Obj Rep 525d1
Coll of Obj Rep 525cn
Coll of Obj Rep 525c1
Coll of Obj Rep 525bn
Coll of Obj Rep 525b1
Coll of Obj Rep 525an
Coll of Obj Rep 525a1
93

FIG. 28



Collection of Sequences
Sequence
Knowledge Cell
530c
533wa
533wb
533wc
533wd
800ca
800cb
800da
800db
800dc
800dd
800de
Decision -making Unit
540
125
Object Processing Unit
93
Coll of Obj Rep
Coll of Obj Rep 525a1
Coll of Obj Rep 525an
Coll of Obj Rep 525b1
Coll of Obj Rep 525bn
Coll of Obj Rep 525c1
Coll of Obj Rep 525cn
Coll of Obj Rep 525d1
Coll of Obj Rep 525dn
Coll of Obj Rep 525e1
Coll of Obj Rep 525en

FIG. 29

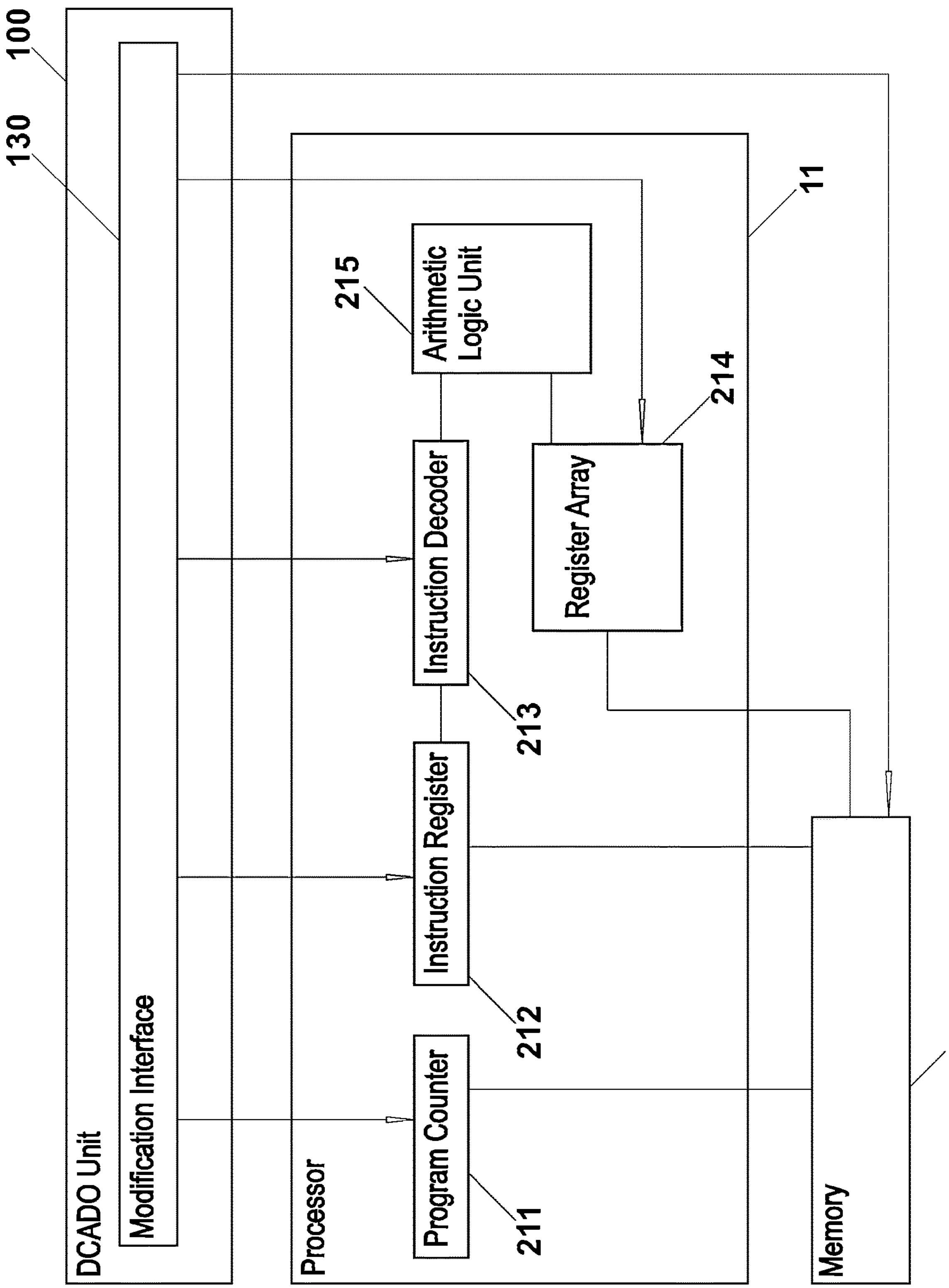

100
130
DCADO Unit
Modification Interface
Processor
11
Program Counter
211
Instruction Register
212
Instruction Decoder
213
Register Array
214
Arithmetic Logic Unit
215
Memory
12

FIG. 30

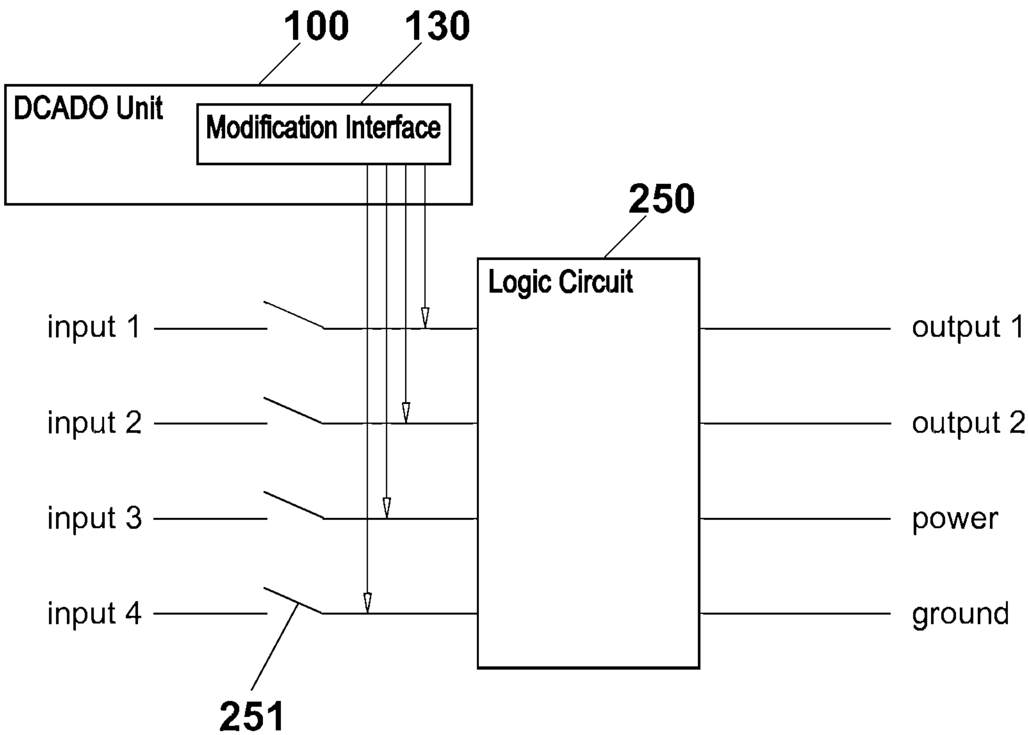


FIG. 31A

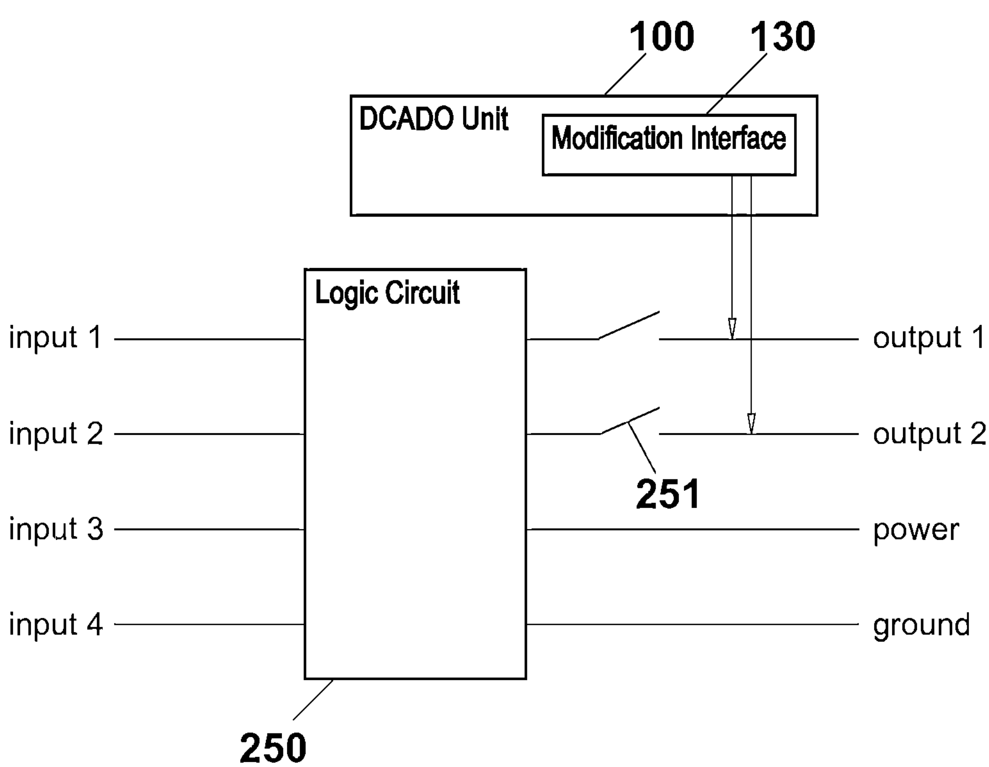


FIG. 31B

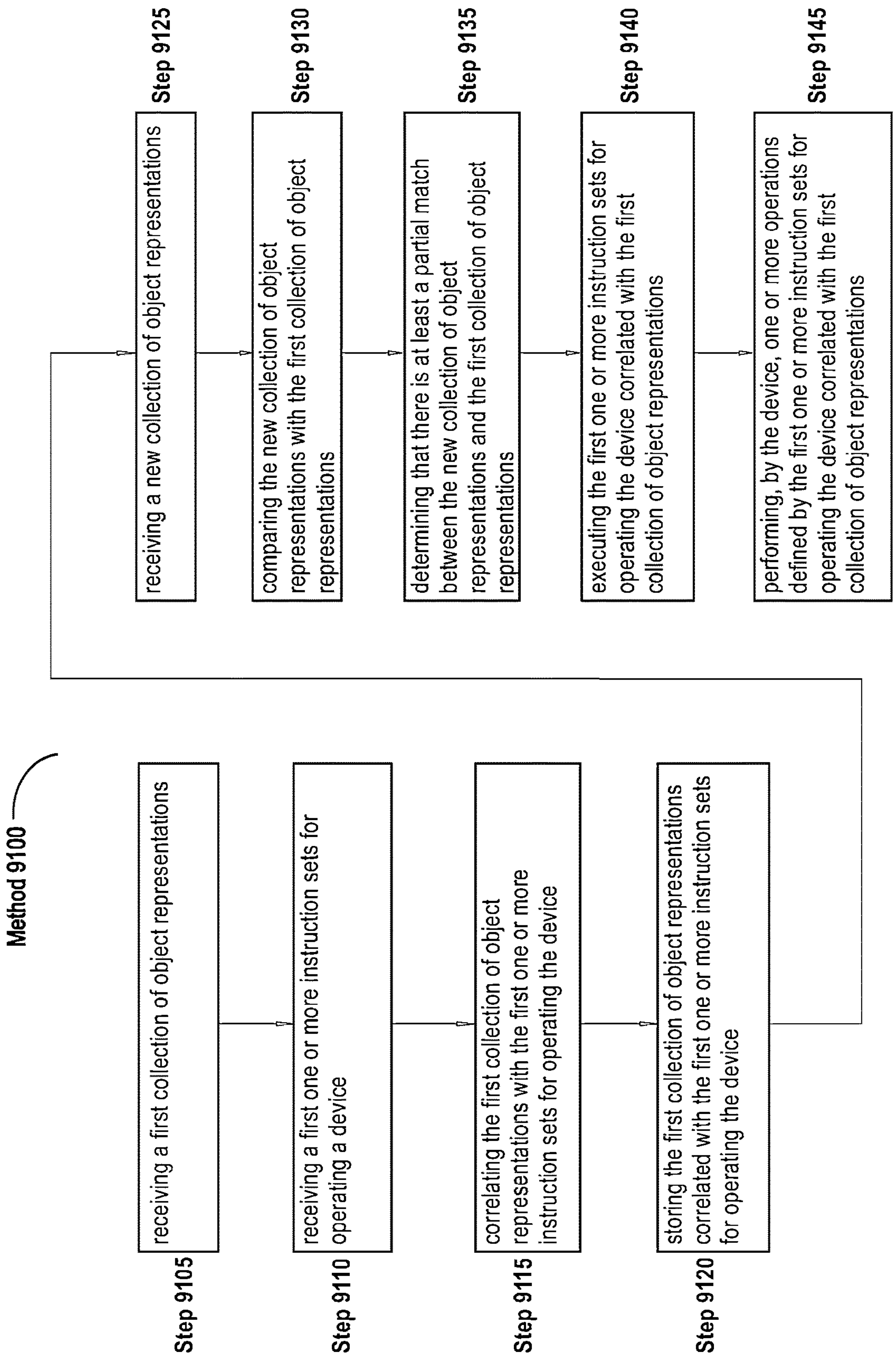

Method 9100
Step 9105
receiving a first collection of object representations
Step 9110
receiving a first one or more instruction sets for operating a device
Step 9115
correlating the first collection of object representations with the first one or more instruction sets for operating the device
Step 9120
storing the first collection of object representations correlated with the first one or more instruction sets for operating the device
Step 9125
receiving a new collection of object representations
Step 9130
comparing the new collection of object representations with the first collection of object representations
Step 9135
determining that there is at least a partial match between the new collection of object representations and the first collection of object representations
Step 9140
executing the first one or more instruction sets for operating the device correlated with the first collection of object representations
Step 9145
performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations

FIG. 32

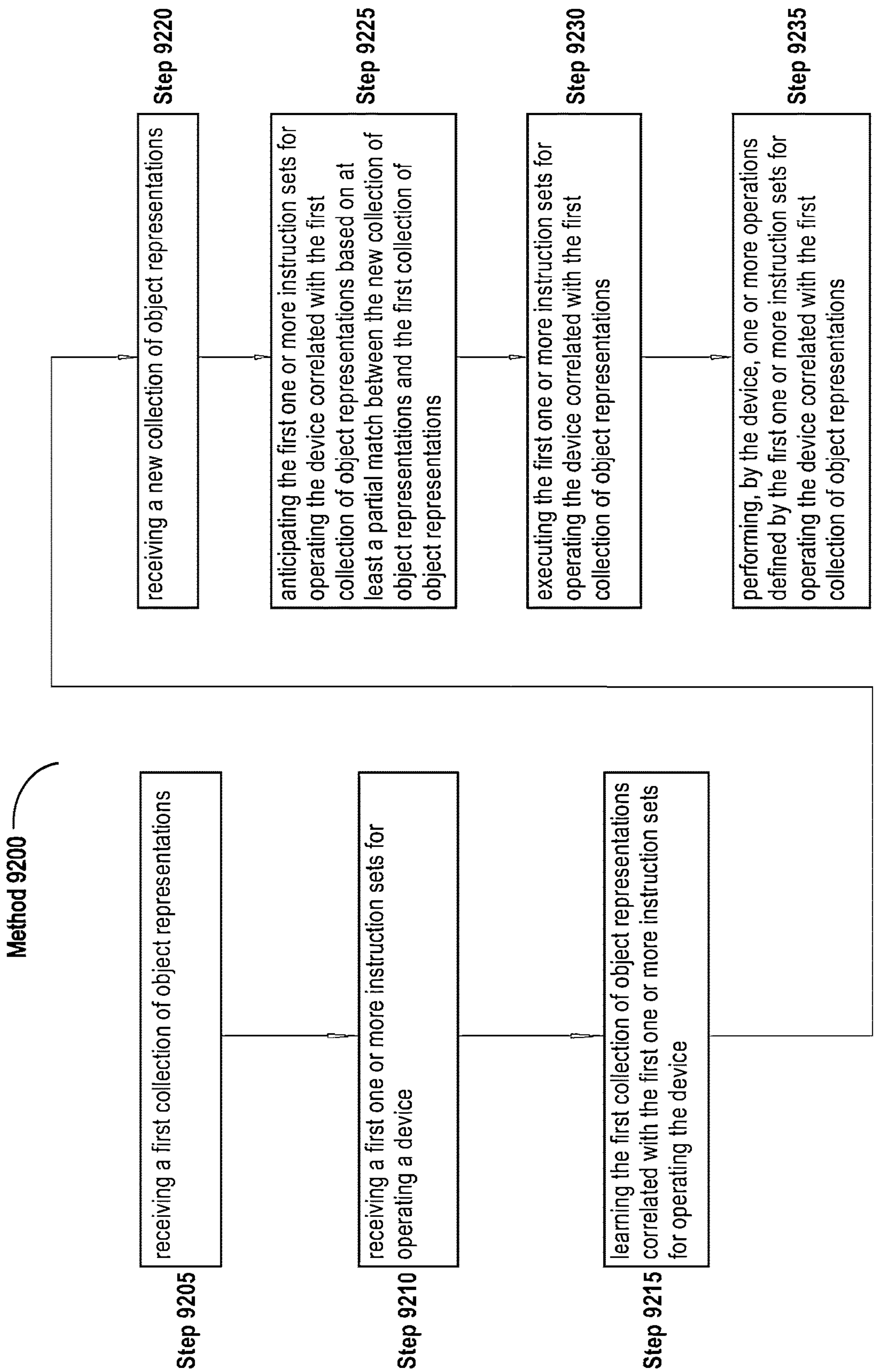

Method 9200
receiving a first collection of object representations
Step 9205
receiving a first one or more instruction sets for operating a device
Step 9210
learning the first collection of object representations correlated with the first one or more instruction sets for operating the device
Step 9215
receiving a new collection of object representations
Step 9220
anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations
Step 9225
executing the first one or more instruction sets for operating the device correlated with the first collection of object representations
Step 9230
performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations
Step 9235

FIG. 33

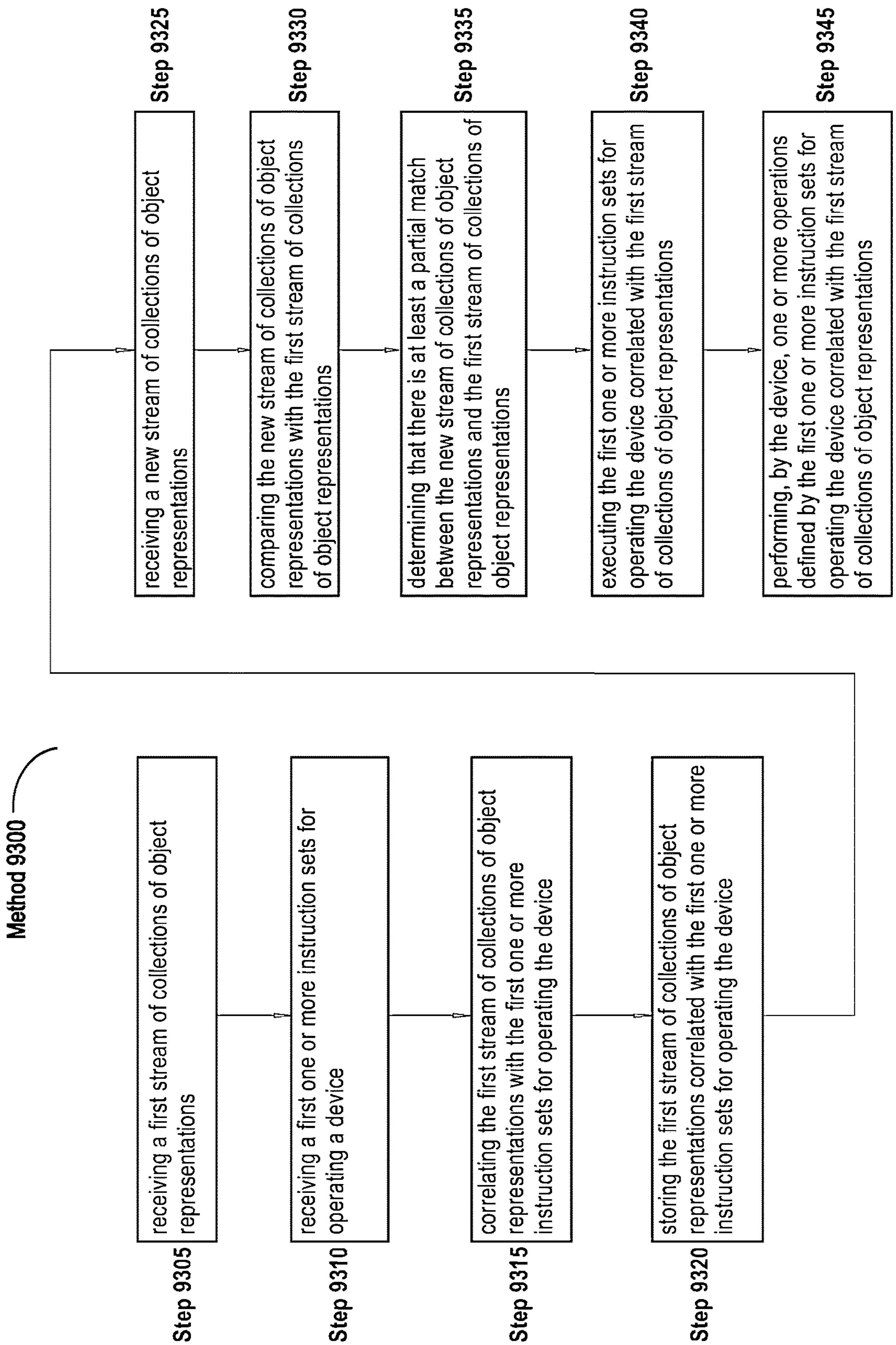

Method 9300
Step 9305
receiving a first stream of collections of object representations
Step 9310
receiving a first one or more instruction sets for operating a device
Step 9315
correlating the first stream of collections of object representations with the first one or more instruction sets for operating the device
Step 9320
storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device
Step 9325
receiving a new stream of collections of object representations
Step 9330
comparing the new stream of collections of object representations with the first stream of collections of object representations
Step 9335
determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations
Step 9340
executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations
Step 9345
performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations

FIG. 34

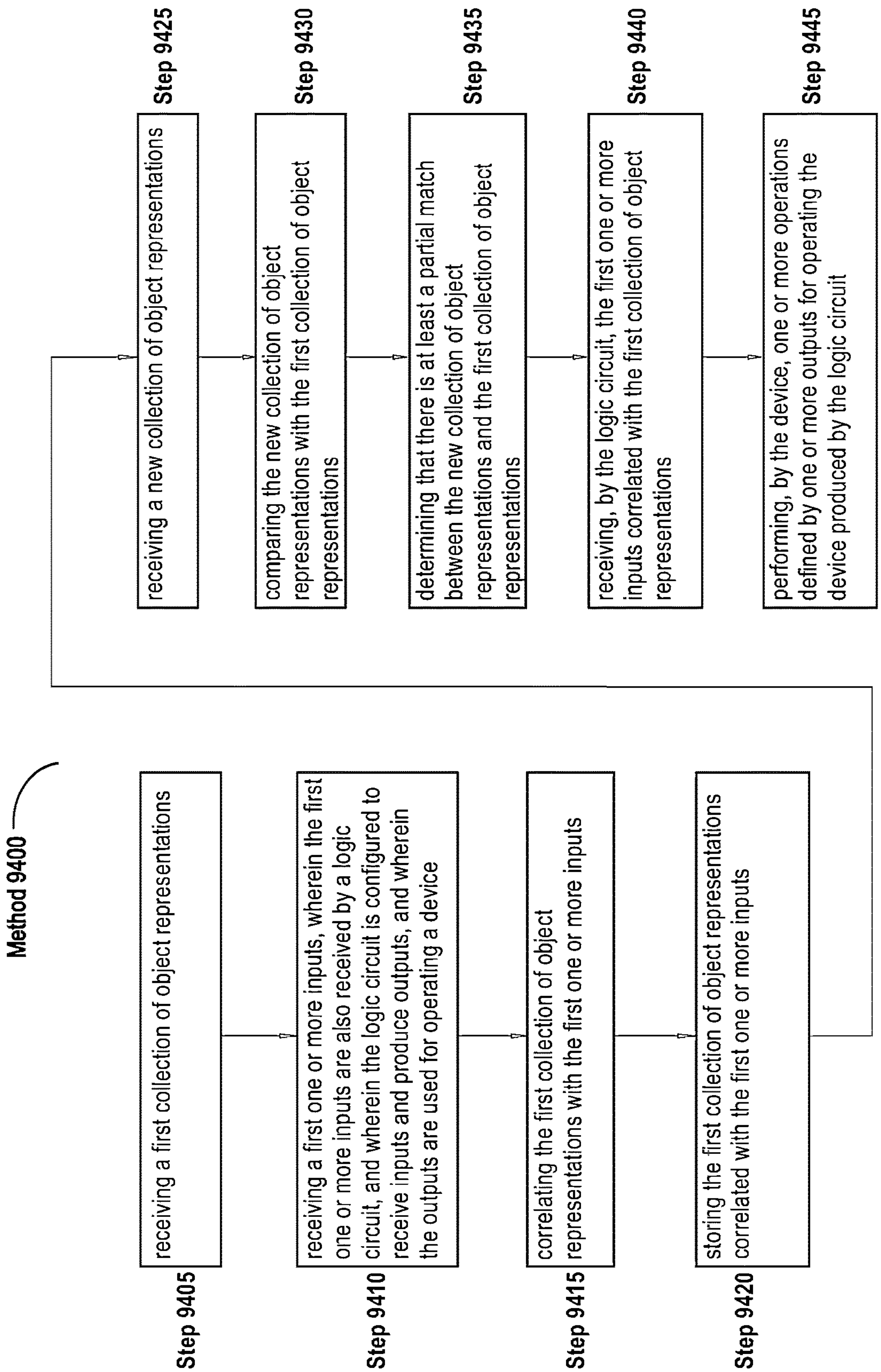

Method 9400
Step 9405
receiving a first collection of object representations
Step 9410
receiving a first one or more inputs, wherein the first one or more inputs are also received by a logic circuit, and wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device
Step 9415
correlating the first collection of object representations with the first one or more inputs
Step 9420
storing the first collection of object representations correlated with the first one or more inputs
Step 9425
receiving a new collection of object representations
Step 9430
comparing the new collection of object representations with the first collection of object representations
Step 9435
determining that there is at least a partial match between the new collection of object representations and the first collection of object representations
Step 9440
receiving, by the logic circuit, the first one or more inputs correlated with the first collection of object representations
Step 9445
performing, by the device, one or more operations defined by one or more outputs for operating the device produced by the logic circuit

FIG. 35

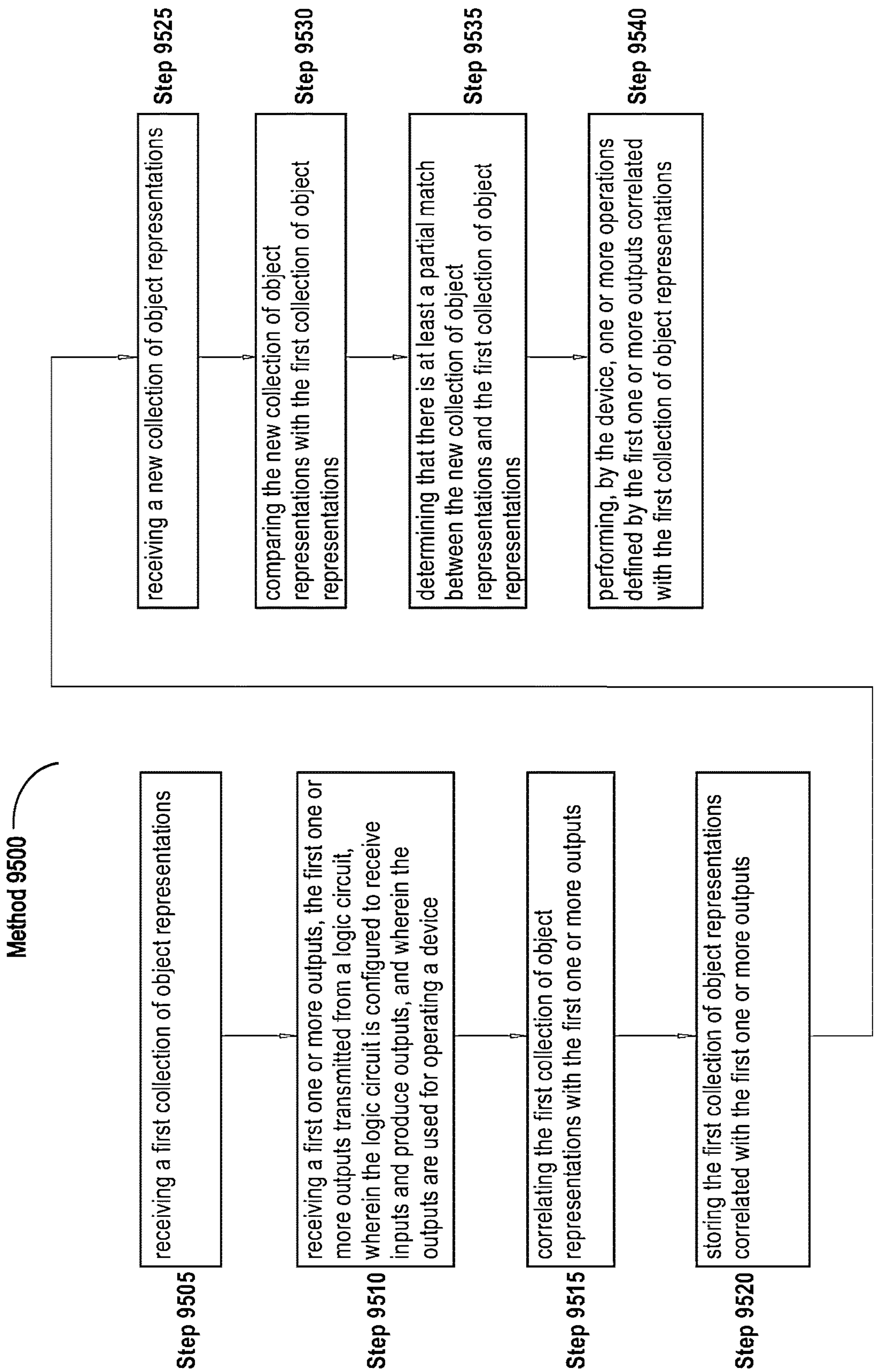

Method 9500
receiving a first collection of object representations
Step 9505
receiving a first one or more outputs, the first one or more outputs transmitted from a logic circuit, wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device
Step 9510
correlating the first collection of object representations with the first one or more outputs
Step 9515
storing the first collection of object representations correlated with the first one or more outputs
Step 9520
receiving a new collection of object representations
Step 9525
comparing the new collection of object representations with the first collection of object representations
Step 9530
determining that there is at least a partial match between the new collection of object representations and the first collection of object representations
Step 9535
performing, by the device, one or more operations defined by the first one or more outputs correlated with the first collection of object representations
Step 9540

FIG. 36

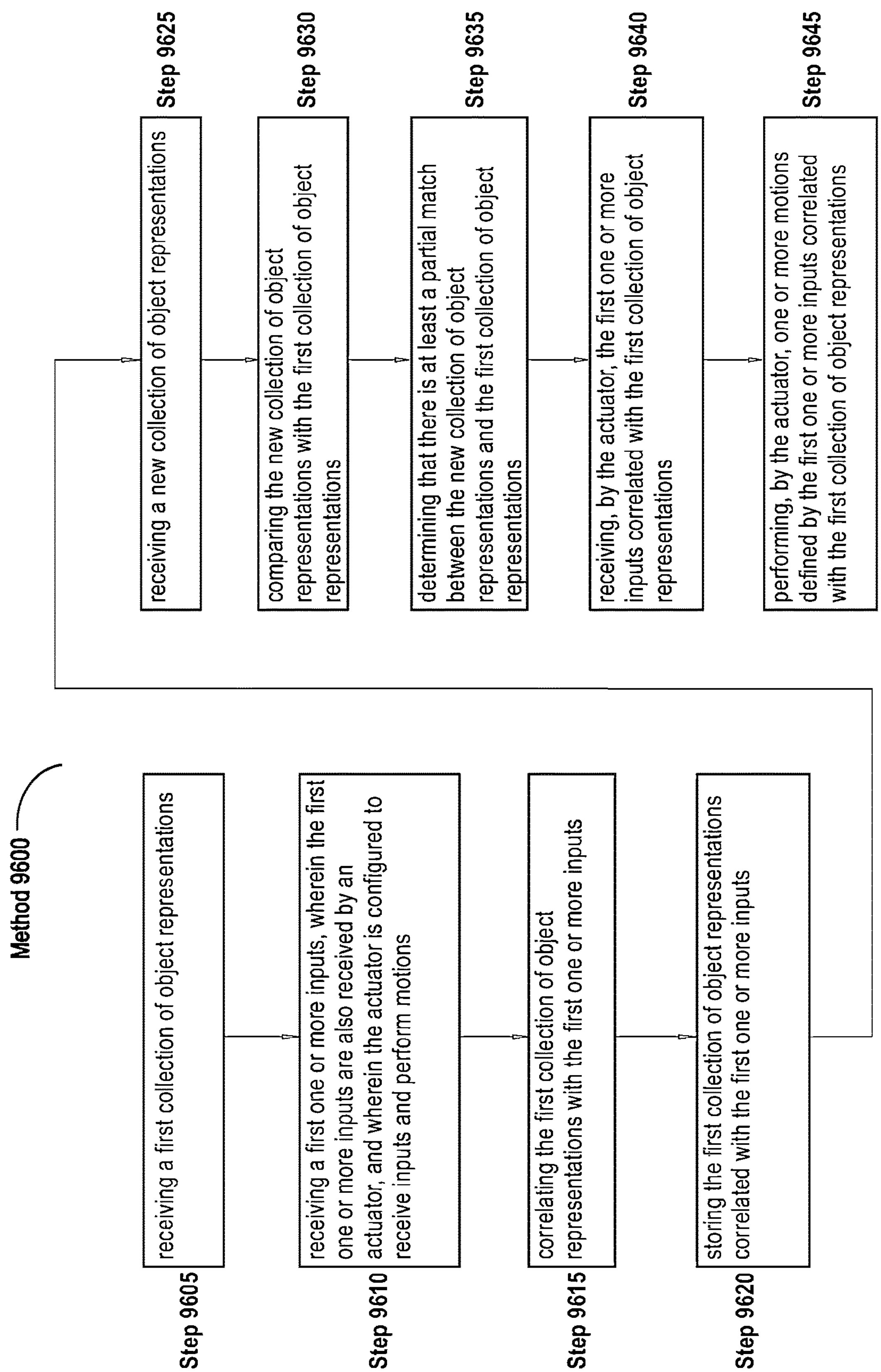

Method 9600
Step 9605
receiving a first collection of object representations
Step 9610
receiving a first one or more inputs, wherein the first one or more inputs are also received by an actuator, and wherein the actuator is configured to receive inputs and perform motions
Step 9615
correlating the first collection of object representations with the first one or more inputs
Step 9620
storing the first collection of object representations correlated with the first one or more inputs
Step 9625
receiving a new collection of object representations
Step 9630
comparing the new collection of object representations with the first collection of object representations
Step 9635
determining that there is at least a partial match between the new collection of object representations and the first collection of object representations
Step 9640
receiving, by the actuator, the first one or more inputs correlated with the first collection of object representations
Step 9645
performing, by the actuator, one or more motions defined by the first one or more inputs correlated with the first collection of object representations

FIG. 37

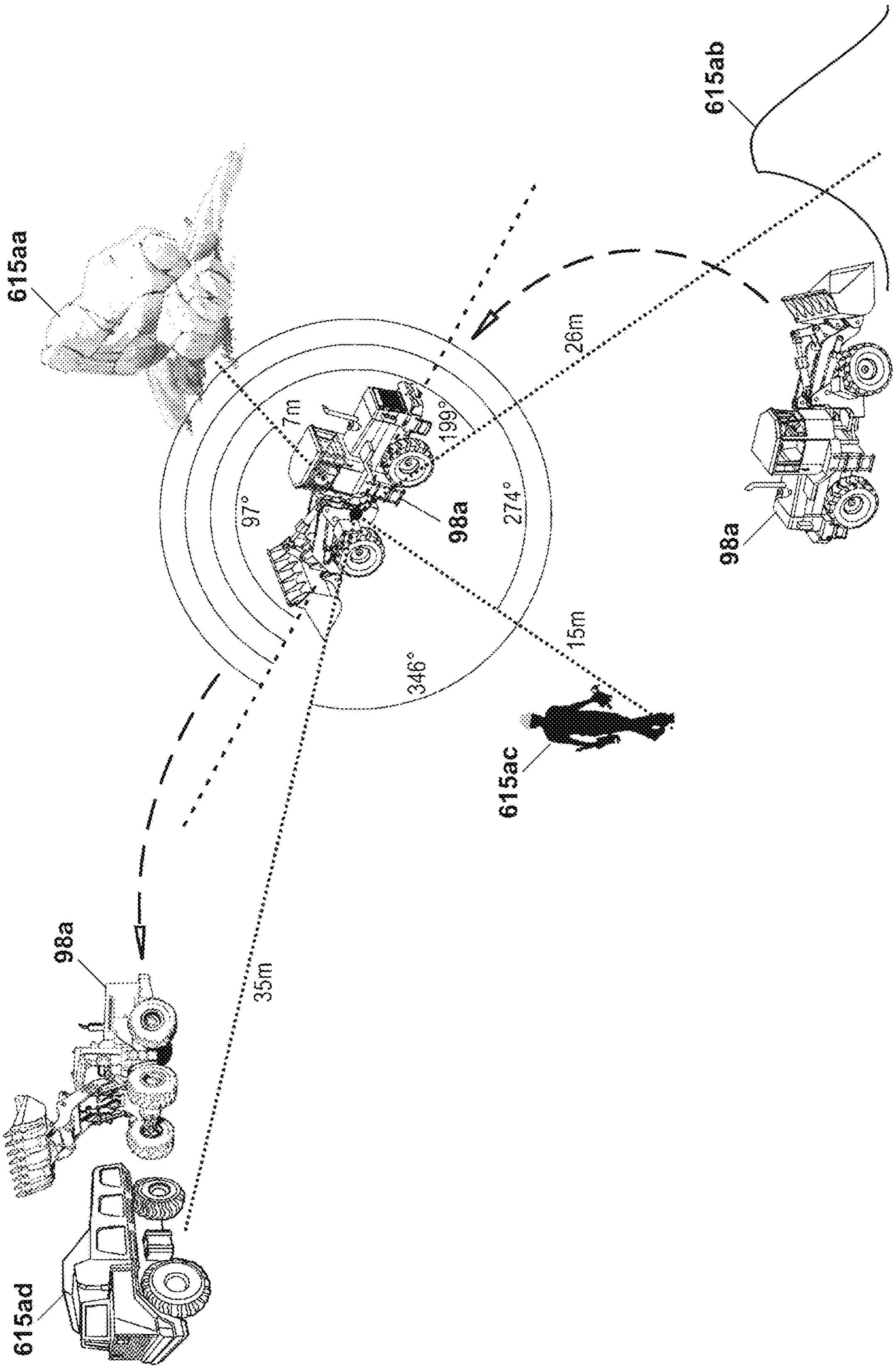

615aa
615ab
26m
7m
199°
97°
98a
274°
98a
15m
346°
615ac
98a
35m
615ad

FIG. 38

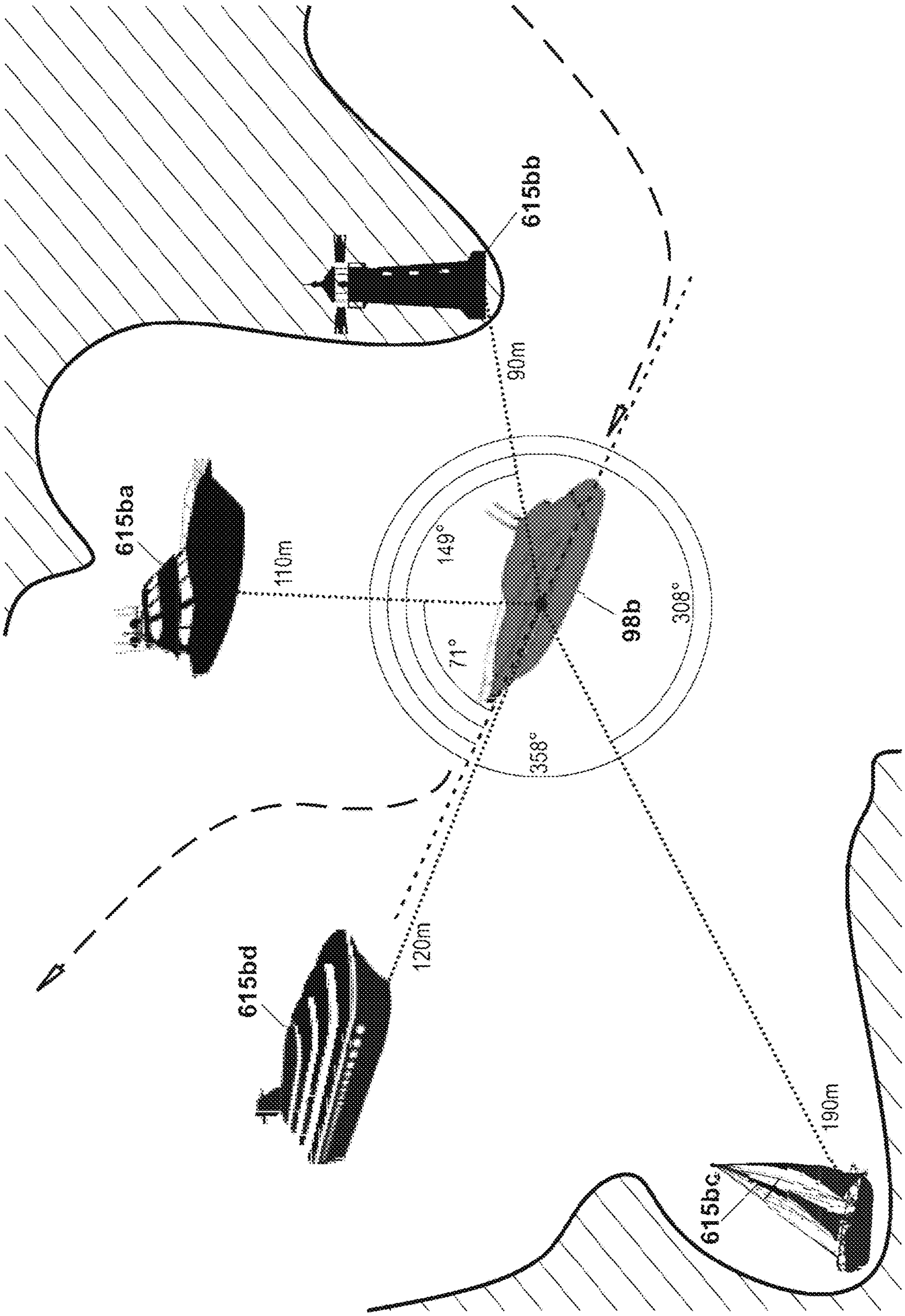


FIG. 39

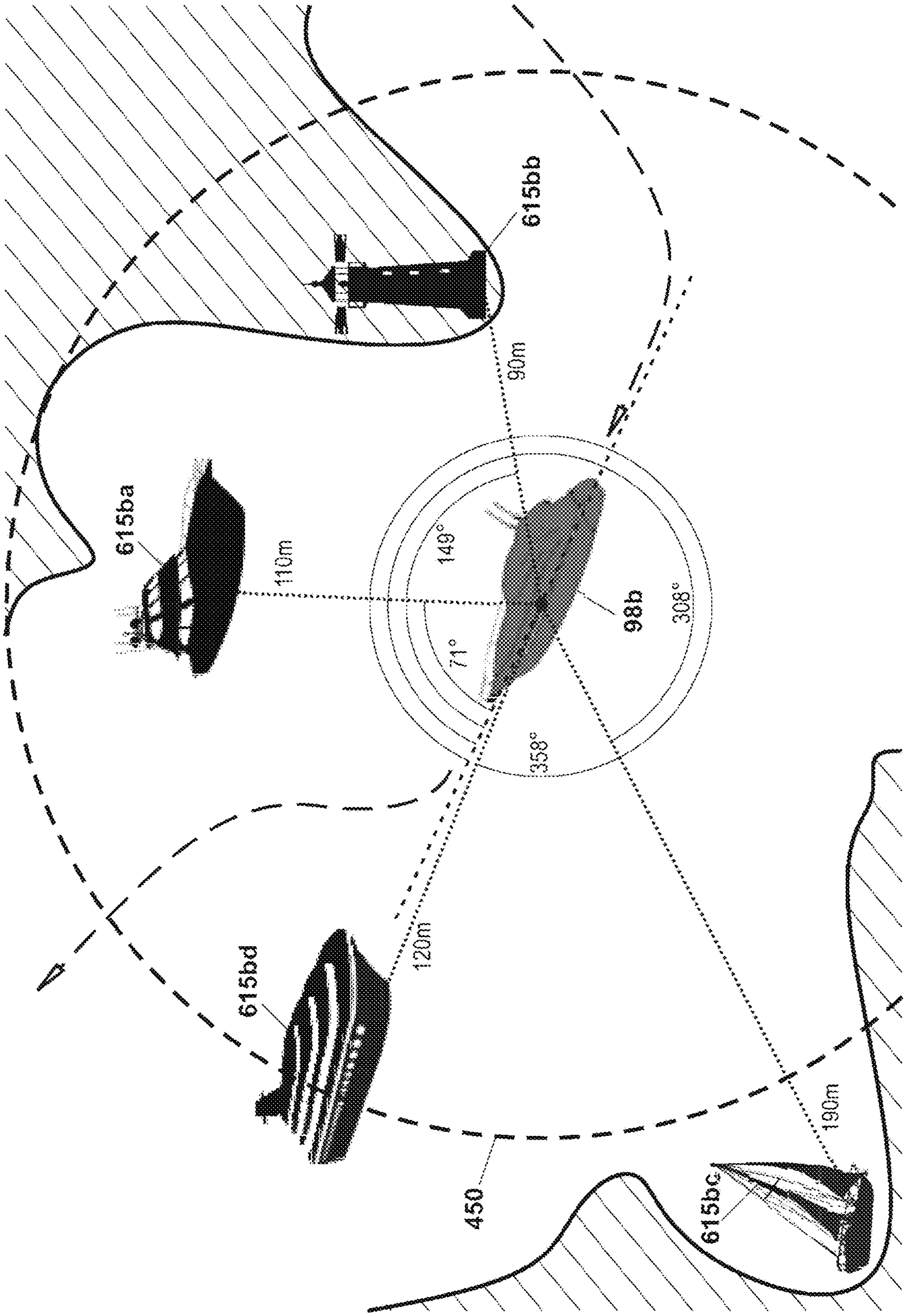

615bb
90m
615ba
110m
149°
98b
308°
71°
358°
120m
615bd
450
190m
615bc

FIG. 40

ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION

CROSS REFERENCE TO RELATED APPLICATION

This application is a continuation of, and claims priority under 35 U.S.C. § 120 from, nonprovisional U.S. patent application Ser. No. 16/540,972 entitled "ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION", filed on Aug. 14, 2019, which is a continuation of, and claims priority under 35 U.S.C. § 120 from, nonprovisional U.S. patent application Ser. No. 15/340,991 entitled "ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION", issued as U.S. Pat. No. 10,452,974, filed on Nov. 2, 2016. The disclosures of the foregoing documents are incorporated herein by reference.

FIELD

The disclosure generally relates to computing enabled devices and/or systems. The disclosure includes devices, apparatuses, systems, and related methods for providing advanced learning, anticipating, decision making, automation, and/or other functionalities.

COPYRIGHT NOTICE

BACKGROUND

Devices or systems commonly operate by receiving a user's operating directions in various circumstances. Instructions are then executed to effect the operation of a device or system based on user's operating directions. Hence, devices or systems rely on the user to direct their behaviors. Commonly employed device or system operating techniques lack a way to learn operation of a device or system and enable autonomous operation of a device or system.

SUMMARY

In some aspects, the disclosure relates to a system for learning and using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises a processor circuit configured to execute instruction sets for operating a device. The system may further include a memory unit configured to store data. The system may further include a sensor configured to detect objects. The system may further include an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first collection of object representations, the first collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to learn the first collection of object representations correlated with the first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: receive a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations performed in response to the executing by the processor circuit.

In some embodiments, at least one of: the processor circuit, the memory unit, the sensor, or the artificial intelligence unit are part of, operating on, or coupled to the device. In further embodiments, the device includes one or more devices. In further embodiments, the device includes a construction machine, an assembly machine, an object handling machine, an object dispensing machine, a sorting machine, a restocking machine, an industrial machine, an agricultural machine, a harvesting machine, an appliance, a toy, a robot, a ground vehicle, an aerial vehicle, an aquatic vehicle, a computer, a smartphone, a control device, or a computing enabled device. In further embodiments, the processor circuit includes one or more processor circuits. In further embodiments, the processor circuit includes a logic circuit. The logic circuit may include a microcontroller. The one or more instruction sets may include one or more inputs into or one or more outputs from the logic circuit.

In certain embodiments, the processor circuit includes a logic circuit, the instruction sets for operating the device include inputs into the logic circuit, and executing instruction sets for operating the device includes performing logic operations on the inputs into the logic circuit and producing outputs for operating the device. The logic circuit includes a microcontroller.

In some embodiments, the processor circuit includes a logic circuit, the instruction sets for operating the device include outputs from the logic circuit for operating the device, and executing instruction sets for operating the device includes performing logic operations on inputs into the logic circuit and producing the outputs from the logic circuit for operating the device.

In certain embodiments, the memory unit includes one or more memory units. In further embodiments, the memory unit resides on a remote computing device or a remote computing system, the remote computing device or the remote computing system coupled to the processor circuit via a network or an interface. The remote computing device or the remote computing system may include a server, a

cloud, a computing device, or a computing system accessible over the network or the interface.

In some embodiments, the sensor includes one or more sensors. In further embodiments, the sensor includes a camera, a microphone, a lidar, a radar, a sonar, or a detector. In further embodiments, the sensor is part of a remote device. In further embodiments, the sensor is configured to detect objects in the device's surrounding.

In certain embodiments, the artificial intelligence unit is coupled to the sensor. In further embodiments, the artificial intelligence unit is coupled to the memory unit. In further embodiments, the artificial intelligence unit is part of, operating on, or coupled to the processor circuit. In further embodiments, the system further comprises: an additional processor circuit, wherein the artificial intelligence unit is part of, operating on, or coupled to the additional processor circuit. In further embodiments, the artificial intelligence unit is part of, operating on, or coupled to a remote computing device or a remote computing system, the remote computing device or the remote computing system coupled to the processor circuit via a network or an interface. In further embodiments, the artificial intelligence unit includes a circuit, a computing apparatus, or a computing system attachable to the processor circuit. In further embodiments, the artificial intelligence unit includes a circuit, a computing apparatus, or a computing system attachable to the device. In further embodiments, the artificial intelligence unit is attachable to an application for operating the device, the application running on the processor circuit. In further embodiments, the artificial intelligence unit includes a circuit, a computing apparatus, or a computing system built into the processor circuit. In further embodiments, the artificial intelligence unit includes a circuit, a computing apparatus, or a computing system built into the device. In further embodiments, the artificial intelligence unit is built into an application for operating the device, the application running on the processor circuit. In further embodiments, the artificial intelligence unit is provided as a feature of the processor circuit. In further embodiments, the artificial intelligence unit is provided as a feature of an application running on the processor circuit. In further embodiments, the artificial intelligence unit is provided as a feature of the device. In further embodiments, the artificial intelligence unit is further configured to: take control from, share control with, or release control to the processor circuit. In further embodiments, the artificial intelligence unit is further configured to: take control from, share control with, or release control to an application or an object of the application, the application running on the processor circuit.

In some embodiments, the first collection of object representations includes one or more representations of objects detected by the sensor at a time. In further embodiments, the new collection of object representations includes one or more representations of objects detected by the sensor at a time. In further embodiments, the first collection of object representations includes a stream of collections of object representations. In further embodiments, the new collection of object representations includes a stream of collections of object representations. In further embodiments, the first or the new collection of object representations includes or is associated with a time stamp, an order, or a time related information. In further embodiments, the first or the new collection of object representations includes one or more representations of objects in the device's surrounding. In further embodiments, the first or the new collection of object representations includes one or more representations of objects in a remote device's surrounding. In further embodiments, an object representation of the one or more object representations includes one or more object properties. In further embodiments, the first or the new collection of object representations includes one or more object properties. In further embodiments, the first collection of object representations includes a comparative collection of object representations whose at least one portion can be used for comparisons with at least one portion of collections of object representations subsequent to the first collection of object representations, the collections of object representations subsequent to the first collection of object representations comprising the new collection of object representations. In further embodiments, the first collection of object representations includes a comparative collection of object representations that can be used for comparisons with the new collection of object representations. In further embodiments, the new collection of object representations includes an anticipatory collection of object representations whose correlated one or more instruction sets can be used for anticipation of one or more instruction sets to be executed by the processor circuit.

In certain embodiments, the first one or more instruction sets for operating the device include one or more instruction sets that temporally correspond to the first collection of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first collection of object representations include one or more instruction sets executed at a time of generating the first collection of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first collection of object representations include one or more instruction sets executed prior to generating the first collection of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first collection of object representations include one or more instruction sets executed within a threshold period of time prior to generating the first collection of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first collection of object representations include one or more instruction sets executed subsequent to generating the first collection of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first collection of object representations include one or more instruction sets executed within a threshold period of time subsequent to generating the first collection of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first collection of object representations include one or more instruction sets executed within a threshold period of time prior to generating the first collection of object representations or a threshold period of time subsequent to generating the first collection of object representations.

In some embodiments, the first one or more instruction sets for operating the device are executed by the processor circuit. In further embodiments, the first one or more instruction sets for operating the device are part of an application for operating the device, the application running on the processor circuit. In further embodiments, the first one or more instruction sets for operating the device include one or more inputs into or one or more outputs from the processor circuit. In further embodiments, the first one or more instruction sets for operating the device include values or states of one or more registers or elements of the processor circuit. In further embodiments, the first one or more instruction sets for operating the device include at least one of: a command, a keyword, a symbol, an instruction, an operator, a variable,

a value, an object, a data structure, a function, a parameter, a state, a signal, an input, an output, a character, a digit, or a reference thereto. In further embodiments, the first one or more instruction sets for operating the device include a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, a structured query language (SQL) code, or a machine code. In further embodiments, the first one or more instruction sets for operating the device include one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the processor circuit includes a logic circuit. The first one or more instruction sets for operating the device may include one or more inputs into a logic circuit. The first one or more instruction sets for operating the device may include one or more outputs from a logic circuit.

In certain embodiments, the first one or more instruction sets for operating the device include one or more instruction sets for operating an application or an object of the application, the application running on the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes obtaining the first one or more instruction sets for operating the device from the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device as they are executed by the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from a register or an element of the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from an element that is part of, operating on, or coupled to the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from at least one of: the memory unit, the device, a virtual machine, a runtime engine, a hard drive, a storage device, a peripheral device, a network connected device, or a user. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from a plurality of processor circuits, applications, memory units, devices, virtual machines, runtime engines, hard drives, storage devices, peripheral devices, network connected devices, or users.

In some embodiments, the processor circuit includes a logic circuit, and wherein the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from the logic circuit. In further embodiments, the logic circuit includes a microcontroller. In further embodiments, the receiving the first one or more instruction sets for operating the device from the logic circuit includes receiving the first one or more instruction sets for operating the device from an element of the logic circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device from the logic circuit includes receiving one or more inputs into the logic circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device from the logic circuit includes receiving one or more outputs from the logic circuit.

In certain embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from an application for operating the device, the application running on the processor circuit.

In some embodiments, the system further comprises: an application including instruction sets for operating the device, the application running on the processor circuit, wherein the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from the application.

In certain embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, a structured query language (SQL) code, or a machine code. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of an element that is part of, operating on, or coupled to the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a register of the processor circuit, the memory unit, a storage, or a repository where the first one or more instruction sets for operating the device are stored. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of the processor circuit, the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a processing element. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of the processor circuit or tracing, profiling, or instrumentation of a component of the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of an application or an object of the application, the application running on the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of one or more of code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a user input. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: a manual, an automatic, a dynamic, or a just in time (JIT) tracing, profiling, or instrumentation. In further embodiments, the receiving the first one or more instruction sets for operating the device includes utilizing at least one of: a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, a logging tool, or an independent tool for obtaining instruction sets. In further embodiments, the

receiving the first one or more instruction sets for operating the device includes utilizing an assembly language. In further embodiments, the receiving the first one or more instruction sets for operating the device includes utilizing a branch or a jump. In further embodiments, the receiving the first one or more instruction sets for operating the device includes a branch tracing or a simulation tracing.

In further embodiments, the system further comprises: an interface configured to receive instruction sets, wherein the first one or more instruction sets for operating the device are received by the interface. The interface may include an acquisition interface.

In some embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the device include a knowledge of how the device operated in a circumstance. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the device are included in a neuron, a node, a vertex, or an element of a knowledgebase. In further embodiments, the knowledgebase includes a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. In further embodiments, some of the neurons, nodes, vertices, or elements are interconnected. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the device are structured into a knowledge cell. In further embodiments, the knowledge cell is included in a neuron, a node, a vertex, or an element of a knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device includes correlating the first collection of object representations with the first one or more instruction sets for operating the device. In further embodiments, the correlating the first collection of object representations with the first one or more instruction sets for operating the device includes generating a knowledge cell, the knowledge cell comprising the first collection of object representations correlated with the first one or more instruction sets for operating the device. In further embodiments, the correlating the first collection of object representations with the first one or more instruction sets for operating the device includes structuring a knowledge of how the device operated in a circumstance. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device includes learning a user's knowledge, style, or methodology of operating the device in a circumstance.

In certain embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device includes storing the first collection of object representations correlated with the first one or more instruction sets for operating the device into the memory unit, the first collection of object representations correlated with the first one or more instruction sets for operating the device being part of a plurality of collections of object representations correlated with one or more instruction sets for operating the device stored in the memory unit. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the device include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the device are organized into a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, each of the plurality of collections of object representations correlated with one or more instruction sets for operating the device is included in a neuron, a node, a vertex, or an element of a knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the device include a user's knowledge, style, or methodology of operating the device in circumstances. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the device are stored on a remote computing device or a remote computing system. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the device include an artificial intelligence system for knowledge structuring, storing, or representation. The artificial intelligence system for knowledge structuring, storing, or representation may include at least one of: a deep learning system, a supervised learning system, an unsupervised learning system, a neural network, a search-based system, an optimization-based system, a logic-based system, a fuzzy logic-based system, a tree-based system, a graph-based system, a hierarchical system, a symbolic system, a sub-symbolic system, an evolutionary system, a genetic system, a multi-agent system, a deterministic system, a probabilistic system, or a statistical system.

In some embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes comparing at least one portion of the new collection of object representations with at least one portion of the first collection of object representations. In further embodiments, the at least one portion of the new collection of object representations include at least one object representation or at least one object property of the new collection of object representations. In further embodiments, the at least one portion of the first collection of object representations include at least one object representation or at least one object property of the first collection of object representations. In further embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes comparing at least one object representation from the new collection of object representations with at least one object representation from the first collection of object representations. In further embodiments, the comparing at least one object representation from the new collection of object representations with at least one object representation from the first collection of object representations includes comparing at least one object property of the at least one object representation from the new collection of

object representations with at least one object property of the at least one object representation from the first collection of object representations.

In certain embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes determining that there is at least a partial match between the new collection of object representations and the first collection of object representations. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that there is at least a partial match between one or more portions of the new collection of object representations and one or more portions of the first collection of object representations. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that a similarity between at least one portion of the new collection of object representations and at least one portion of the first collection of object representations exceeds a similarity threshold. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining a substantial similarity between at least one portion of the new collection of object representations and at least one portion of the first collection of object representations. The substantial similarity may be achieved when a similarity between the at least one portion of the new collection of object representations and the at least one portion of the first collection of object representations exceeds a similarity threshold. The substantial similarity may be achieved when a number or a percentage of matching or partially matching portions of the new collection of object representations and portions of the first collection of object representations exceeds a threshold number or threshold percentage. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that a number or a percentage of matching or partially matching object representations from the new collection of object representations and from the first collection of object representations exceeds a threshold number or threshold percentage. The matching or partially matching object representations from the new collection of object representations and from the first collection of object representations may be determined factoring in at least one of: a type of an object representation, an importance of an object representation, a threshold for a similarity in an object representation, or a threshold for a difference in an object representation. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that a number or a percentage of matching or partially matching object properties from the new collection of object representations and from the first collection of object representations exceeds a threshold number or threshold percentage. The matching or partially matching object properties from the new collection of object representations and from the first collection of object representations may be determined factoring in at least one of: an association of an object property with an object representation, a category of an object property, an importance of an object property, a threshold for a similarity in an object property, or a threshold for a difference in an object property. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that there is at least a partial match between at least one object representation from the new collection of object representations and at least one object representation from the first collection of object representations. The determining that there is at least a partial match between at least one object representation from the new collection of object representations and at least one object representation from the first collection of object representations includes determining that there is at least a partial match between at least one object property of the at least one object representation from the new collection of object representations and at least one object property of the at least one object representation from the first collection of object representations.

In certain embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations instead of or prior to an instruction set that would have been executed next. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first collection of object representations into a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting the processor circuit to the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting the processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes transmitting, to the processor circuit for execution, the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes issuing an interrupt to the processor circuit and executing the first one or more instruction sets for operating the device correlated with the first collection of object representations following the interrupt.

In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying an element that is part of, operating on, or coupled to the processor circuit.

In some embodiments, the processor circuit includes a logic circuit, and wherein the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the logic circuit includes a microcontroller. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying an element of the logic circuit. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first collection of object representations into an element of the logic circuit. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting the logic circuit to the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes replacing inputs into the logic circuit with the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes replacing outputs from the logic circuit with the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In certain embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes causing an application for operating the device to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations, the application running on the processor circuit.

In some embodiments, the system further comprises: an application including instruction sets for operating the device, the application running on the processor circuit, wherein the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying the application.

In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting an application to the first one or more instruction sets for operating the device correlated with the first collection of object representations, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting an application to one or more alternate instruction sets, the application running on the processor circuit, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying at least one of: the memory unit, a register of the processor circuit, a storage, or a repository where instruction sets are stored or used. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets for operating an application or an object of the application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying at least one of: an element of the processor circuit, an element of the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of

object representations includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing an assembly language. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes adding or inserting additional code into a code of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes at least one of: modifying, removing, rewriting, or overwriting a code of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of an application, the application running on the processor circuit. The branching or redirecting the code may include inserting at least one of: a branch, a jump, or a means for redirecting an execution. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes implementing a user's knowledge, style, or methodology of operating the device in a circumstance.

In certain embodiments, the system further comprises: an interface configured to cause execution of instruction sets, wherein the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations is caused by the interface. The interface may include a modification interface.

In some embodiments, the one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations include one or more operations with or by a computing enabled device. In further embodiments, the performing the one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations includes implementing a user's knowledge, style, or methodology of operating the device in a circumstance.

In certain embodiments, the system further comprising: an application running on the processor circuit.

In some embodiments, the instruction sets for operating the device are part of an application for operating the device, the application running on the processor circuit.

In certain embodiments, the system of further comprises: an application for operating the device, the application running on the processor circuit. The application for operating the device may include the instruction sets for operating the device.

In some embodiments, the artificial intelligence unit is further configured to: receive at least one extra information. In further embodiments, the at least one extra information include one or more of: a time information, a location information, a computed information, or a contextual information. In further embodiments, the at least one extra information include one or more of: an information on an object, an information on an object representation, an information on a collection of object representations, an information on a device's circumstance, an information on an instruction set, an information on an application, an information on the processor circuit, an information on the device, or an information on an user. In further embodiments, the artificial intelligence unit is further configured to: learn the first collection of object representations correlated with the at least one extra information. The learning the first collection of object representations correlated with at least one extra information may include correlating the first collection of object representations with the at least one extra information. The learning the first collection of object representations correlated with at least one extra information may include storing the first collection of object representations correlated with the at least one extra information into the memory unit. In further embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations. The anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations may include comparing an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations. The anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations may include determining that a similarity between an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations exceeds a similarity threshold.

In certain embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: cause the user interface to present a user with an option to execute the first one or more instruc-

tion sets for operating the device correlated with the first collection of object representations.

In some embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: receive, via the user interface, a user's selection to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In certain embodiments, the artificial intelligence unit is further configured to: rate the executed first one or more instruction sets for operating the device correlated with the first collection of object representations. The rating the executed first one or more instruction sets for operating the device correlated with the first collection of object representations may include causing a user interface to display the executed first one or more instruction sets for operating the device correlated with the first collection of object representations along with one or more rating values as options to be selected by a user. The rating the executed first one or more instruction sets for operating the device correlated with the first collection of object representations may include rating the executed first one or more instruction sets for operating the device correlated with the first collection of object representations without a user input.

In some embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: cause the user interface to present a user with an option to cancel the execution of the executed first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the canceling the execution of the executed first one or more instruction sets for operating the device correlated with the first collection of object representations includes restoring the processor circuit or the device to a prior state. The restoring the processor circuit or the device to a prior state may include saving the state of the processor circuit or the device prior to executing the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In certain embodiments, the system further comprises: an input device configured to receive a user's operating directions, the user's operating directions for instructing the processor circuit on how to operate the device.

In some embodiments, the autonomous device operating includes a partially or a fully autonomous device operating. The partially autonomous device operating may include executing the first one or more instruction sets for operating the device correlated with the first collection of object representations responsive to a user confirmation. In further embodiments, the fully autonomous device operating may include executing the first one or more instruction sets for operating the device correlated with the first collection of object representations without a user confirmation.

In certain embodiments, the artificial intelligence unit is further configured to: receive a second collection of object representations, the second collection of object representations including one or more representations of objects detected by the sensor; receive a second one or more instruction sets for operating the device; and learn the second collection of object representations correlated with the second one or more instruction sets for operating the device. In further embodiments, the second collection of object representations includes one or more representations of objects detected by the sensor at a time. In further embodiments, the second collection of object representations includes a stream of collections of object representations. In further embodiments, the second collection of object representations includes or is associated with a time stamp, an order, or a time related information. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device and the learning the second collection of object representations correlated with the second one or more instruction sets for operating the device include creating a connection between the first collection of object representations correlated with the first one or more instruction sets for operating the device and the second collection of object representations correlated with the second one or more instruction sets for operating the device. The connection may include or be associated with at least one of: an occurrence count, a weight, a parameter, or a data. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device and the learning the second collection of object representations correlated with the second one or more instruction sets for operating the device include updating a connection between the first collection of object representations correlated with the first one or more instruction sets for operating the device and the second collection of object representations correlated with the second one or more instruction sets for operating the device. The updating the connection between the first collection of object representations correlated with the first one or more instruction sets for operating the device and the second collection of object representations correlated with the second one or more instruction sets for operating the device may include updating at least one of: an occurrence count, a weight, a parameter, or a data included in or associated with the connection. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device includes storing the first collection of object representations correlated with the first one or more instruction sets for operating the device into a first node of a knowledgebase, and wherein the learning the second collection of object representations correlated with the second one or more instruction sets for operating the device includes storing the second collection of object representations correlated with the second one or more instruction sets for operating the device into a second node of the knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. The knowledgebase may be stored in the memory unit. The learning the first collection of object representations correlated with the first one or more instruction sets for operating the device and the learning the second collection of object representations correlated with the second one or more instruction sets for operating the device include creating a connection between the first node and the second node. The learning the first collection of object representations correlated with the first one or more instruction sets for operating the device and the learning the second collection of object representations correlated with the second one or more instruction sets for operating the device include updating a connection between the first node and the second node. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the device is stored into a first node of a neural network and the second collection of object representations correlated with the second one or more instruction sets for operating the device is stored into a second node of the neural network. The first node and the second node may be connected by a connection. The first

node may be part of a first layer of the neural network and the second node may be part of a second layer of the neural network. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the device is stored into a first node of a graph and the second collection of object representations correlated with the second one or more instruction sets for operating the device is stored into a second node of the graph. The first node and the second node may be connected by a connection. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the device is stored into a first node of a sequence and the second collection of object representations correlated with the second one or more instruction sets for operating the device is stored into a second node of the sequence.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first collection of object representations, the first collection of object representations including one or more representations of objects detected by a sensor. The operations may further include: receiving a first one or more instruction sets for operating a device. The operations may further include: learning the first collection of object representations correlated with the first one or more instruction sets for operating the device. The operations may further include: receiving a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The operations may further include: anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further include: causing an execution of the first one or more instruction sets for operating the device correlated with the first collection of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations by the device performed in response to the execution.

In certain embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from the one or more processor circuits or from another one or more processor circuits. In further embodiments, the execution of the first one or more instruction sets for operating the device correlated with the first collection of object representations is performed by the one or more processor circuits or by another one or more processor circuits.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a processor circuit, the first collection of object representations including one or more representations of objects detected by a sensor. The method may further include: (b) receiving a first one or more instruction sets for operating a device by the processor circuit. The method may further include: (c) learning the first collection of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the processor circuit. The method may further include: (d) receiving a new collection of object representations by the processor circuit, the new collection of object representations including one or more representations of objects detected by the sensor. The method may further include: (e) anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (e) performed by the processor circuit. The method may further include: (f) executing the first one or more instruction sets for operating the device correlated with the first collection of object representations, the executing of (f) performed in response to the anticipating of (e). The method may further include: (g) performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations by the device performed in response to the executing of (f).

In certain embodiments, the receiving of (b) includes receiving the first one or more instruction sets for operating the device from the processor circuit or from another processor circuit. In further embodiments, the executing of (f) is performed by the processor circuit or by another processor circuit.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable as well as the following embodiments.

In some embodiments, the first one or more instruction sets for operating the device are executed by a processor circuit. In further embodiments, the first one or more instruction sets for operating the device are part of an application for operating the device. In further embodiments, the first one or more instruction sets for operating the device include one or more inputs into or one or more outputs from a processor circuit. In further embodiments, the first one or more instruction sets for operating the device include values or states of one or more registers or elements of a processor circuit. In further embodiments, the first one or more instruction sets for operating the device include one or more inputs into a logic circuit. In further embodiments, the first one or more instruction sets for operating the device include one or more outputs from a logic circuit. In further embodiments, the first one or more instruction sets for operating the device include one or more instruction sets for operating an application or an object of the application.

In certain embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device as they are executed by a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from a register or an element of a processor circuit. In further embodiments, the receiving the

first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from an element that is part of, operating on, or coupled to a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from at least one of: the memory unit, the device, a virtual machine, a runtime engine, a hard drive, a storage device, a peripheral device, a network connected device, or a user. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from a plurality of processor circuits, applications, memory units, devices, virtual machines, runtime engines, hard drives, storage devices, peripheral devices, network connected devices, or users.

In some embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from a logic circuit. In further embodiments, the logic circuit includes a microcontroller. In further embodiments, the receiving the first one or more instruction sets for operating the device from the logic circuit includes receiving the first one or more instruction sets for operating the device from an element of the logic circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device from the logic circuit includes receiving one or more inputs into the logic circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device from the logic circuit includes receiving one or more outputs from the logic circuit.

In certain embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from an application for operating the device. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from an application, the application including instruction sets for operating the device. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of an element that is part of, operating on, or coupled to a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a register of a processor circuit, the memory unit, a storage, or a repository where the first one or more instruction sets for operating the device are stored. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a processor circuit, the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a processing element. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a processor circuit or tracing, profiling, or instrumentation of a component of a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of an application or an object of the application. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device by an interface. The interface may include an acquisition interface.

In some embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device includes storing the first collection of object representations correlated with the first one or more instruction sets for operating the device into a memory unit, the first collection of object representations correlated with the first one or more instruction sets for operating the device being part of a plurality of collections of object representations correlated with one or more instruction sets for operating the device stored in the memory unit.

In certain embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes executing the first one or more instruction sets for operating the device correlated with the first collection of object representations instead of or prior to an instruction set that would have been executed next. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying a register or an element of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first collection of object representations into a register or an element of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting a processor circuit to the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting a processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes transmitting, to a processor circuit for execution, the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes issuing an interrupt to a processor circuit and executing the first one or more instruction sets for operating the device correlated with the first collection of object representations following the interrupt. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying an element that is part of, operating on, or coupled to a processor circuit.

In some embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes executing, by a logic circuit, the first one or more instruction sets for operating the device correlated with the first collection of

object representations. In further embodiments, the logic circuit includes a microcontroller. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying an element of the logic circuit. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first collection of object representations into an element of the logic circuit. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting the logic circuit to the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations includes replacing inputs into the logic circuit with the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations includes replacing outputs from the logic circuit with the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In certain embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes executing, by an application for operating the device, the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying an application, the application including instruction sets for operating the device. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting an application to the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting an application to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying at least one of: the memory unit, a register of a processor circuit, a storage, or a repository where instruction sets are stored or used. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets for operating an application or an object of the application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying at least one of: an element of a processor circuit, an element of the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing an assembly language. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes adding or inserting additional code into a code of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes at least one of:

modifying, removing, rewriting, or overwriting a code of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of an application. The branching or redirecting the code may include inserting at least one of: a branch, a jump, or a means for redirecting an execution. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes implementing a user's knowledge, style, or methodology of operating the device in a circumstance. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes executing the first one or more instruction sets for operating the device correlated with the first collection of object representations via an interface. The interface may include a modification interface.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving at least one extra information. In further embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: learning the first collection of object representations correlated with the at least one extra information.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: presenting, via a user interface, a user with an option to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving, via a user interface, a user's selection to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: rating the executed first one or more instruction sets for operating the device correlated with the first collection of object representations.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: presenting, via a user interface, a user with an option to cancel the execution of the executed first one or more instruction sets for operating the device correlated with the first collection of object representations.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving, via an input device, a user's operating directions, the user's operating directions for instructing a processor circuit on how to operate the device.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving a second collection of object representations, the second collection of object representations including one or more representations of objects detected by the sensor; receiving a second one or more instruction sets for operating the device; and learning the second collection of object representations correlated with the second one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving, by a first processor circuit of the one or more processor circuits, a first collection of object representations, the first collection of object representations including one or more representations of objects detected by a sensor. The operations may further comprise: receiving, by the first processor circuit of the one or more processor circuits, a first one or more instruction sets for operating a device. The operations may further comprise: learning, by the first processor circuit of the one or more processor circuits, the first collection of object representations correlated with the first one or more instruction sets for operating the device. The operations may further comprise: receiving, by the first processor circuit of the one or more processor circuits, a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The operations may further comprise: anticipating, by the first processor circuit of the one or more processor circuits, the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further comprise: causing, by the first processor circuit of the one or more processor circuits, an execution, by a second processor circuit of the one or more processor circuits, of the first one or more instruction sets for operating the device correlated with the first collection of object representations, the causing performed in response to the anticipating, by the first processor circuit of the one or more processor circuits, the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations by the device performed in response to the execution.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a first processor circuit, the first collection of object representations including one or more representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more instruction sets for operating a device by the first processor circuit. The method may further comprise: (c) learning the first collection of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the first processor circuit. The method may further comprise: (d) receiving a new collection of object representations by the first processor circuit, the new collection of object representations including one or more representations of objects detected by the sensor. The method may further comprise: (e) anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (e) performed by the first processor circuit. The method may further comprise: (f) executing, by a second processor circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations, the executing of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the device, one or more

operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations by the device performed in response to the executing of (f).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for learning a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first collection of object representations, the first collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: learn the first collection of object representations correlated with the first one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first collection of object representations, the first collection of object representations including one or more representations of objects detected by a sensor. The operations may further comprise: receiving a first one or more instruction sets for operating a device. The operations may further comprise: learning the first collection of object representations correlated with the first one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a processor circuit, the first collection of object representations including one or more representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more instruction sets for operating a device by the processor circuit. The method may further comprise: (c) learning the first collection of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the processor circuit.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: access the memory unit that comprises a plurality of collections of object representations correlated with one or more instruction sets for operating the device, the plurality of collections of object representations correlated with one or more instruction sets for operating the device including a first collection of object representations correlated with a first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: receive a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations performed in response to the executing by the processor circuit.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: accessing a memory unit that comprises a plurality of collections of object representations correlated with one or more instruction sets for operating a device, the plurality of collections of object representations correlated with one or more instruction sets for operating the device including a first collection of object representations correlated with a first one or more instruction sets for operating the device. The operations may further comprise: receiving a new collection of object representations, the new collection of object representations including one or more representations of objects detected by a sensor. The operations may further comprise: anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further comprise: causing an execution of the first one or more instruction sets for operating the device correlated with the first collection of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations by the device performed in response to the execution.

In some aspects, the disclosure relates to a method comprising: (a) accessing a memory unit that comprises a plurality of collections of object representations correlated with one or more instruction sets for operating a device, the plurality of collections of object representations correlated with one or more instruction sets for operating the device including a first collection of object representations correlated with a first one or more instruction sets for operating the device, the accessing of (a) performed by a processor circuit. The method may further comprise: (b) receiving a new collection of object representations by the processor circuit, the new collection of object representations including one or more representations of objects detected by a sensor. The method may further comprise: (c) anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (c) performed by the processor circuit. The method may further comprise: (d) executing the first one or more instruction sets for operating the device correlated with the first collection of object representations, the executing of (d) performed in response to the anticipating of (c). The method may further comprise: (e) performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations by the device performed in response to the executing of (d).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for learning and using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first stream of collections of object representations, the first stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: learn the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: receive a new stream of collections of object representations, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations performed in response to the executing by the processor circuit.

In certain embodiments, each collection of object representations includes one or more representations of objects detected by the sensor at a time. In further embodiments, each collection of object representations includes one or more of object representations. In further embodiments, each collection of object representations includes or is associated with a time stamp, an order, or a time related information. In further embodiments, the first stream of collections of object representations includes one or more collections of representations of objects detected by the sensor over time. In further embodiments, the new stream of collections of object representations includes one or more collections of representations of objects detected by the sensor over time. In further embodiments, the first or the new stream of collections of object representations includes one or more collections of representations of objects in the device's surrounding. In further embodiments, the first or the new stream of collections of object representations includes one or more collections of representations of objects in a remote device's surrounding. In further embodiments, an object representation of a stream of collections of object representations includes one or more object properties. In further embodiments, the first or the new stream of collections of object representations includes one or more object properties. In further embodiments, the first stream of collections of object representations includes a comparative stream of collections of object representations whose at least one portion can be used for comparisons with at least one portion of streams of collections of object representations subsequent to the first stream of collections of object representations, the streams of collections of object representations subsequent to the first stream of collections of object representations comprising the new stream of collections of object representations. In further embodiments, the first stream of collections of object representations includes a comparative stream of collections of object representations that can be used for comparisons with the new stream of collections of object representations. In further embodiments, the new stream of collections of object representations includes an anticipatory stream of collections of object representations whose correlated one or more instruction sets can be used for anticipation of one or more instruction sets to be executed by the processor circuit.

In some embodiments, the first one or more instruction sets for operating the device include one or more instruction sets that temporally correspond to the first stream of collections of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first stream of collections of object representations include one or more instruction sets executed at a time of generating the first stream of collections of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first stream of collections of object representations include one or more instruction sets executed prior to generating the first stream of collections of object representations. In further embodi-

ments, the one or more instruction sets that temporally correspond to the first stream of collections of object representations include one or more instruction sets executed within a threshold period of time prior to generating the first stream of collections of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first stream of collections of object representations include one or more instruction sets executed subsequent to generating the first stream of collections of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first stream of collections of object representations include one or more instruction sets executed within a threshold period of time subsequent to generating the first stream of collections of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first stream of collections of object representations include one or more instruction sets executed within a threshold period of time prior to generating the first stream of collections of object representations or a threshold period of time subsequent to generating the first stream of collections of object representations.

In certain embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device include a knowledge of how the device operated in a circumstance. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device are included in a neuron, a node, a vertex, or an element of a knowledgebase. In further embodiments, the knowledgebase includes a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. In further embodiments, some of the neurons, nodes, vertices, or elements are interconnected. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device are structured into a knowledge cell. In further embodiments, the knowledge cell is included in a neuron, a node, a vertex, or an element of a knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device includes correlating the first stream of collections of object representations with the first one or more instruction sets for operating the device. In further embodiments, the correlating the first stream of collections of object representations with the first one or more instruction sets for operating the device includes generating a knowledge cell, the knowledge cell comprising the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device. In further embodiments, the correlating the first stream of collections of object representations with the first one or more instruction sets for operating the device includes structuring a knowledge of how the device operated in a circumstance. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device includes learning a user's knowledge, style, or methodology of operating the device in a circumstance.

In some embodiments, the learning the first stream of collections of object representations correlated with the first

one or more instruction sets for operating the device includes storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device into the memory unit, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device being part of a plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device stored in the memory unit. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device are organized into a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, each stream of collections of object representations correlated with one or more instruction sets for operating the device of the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device is included in a neuron, a node, a vertex, or an element of a knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device include a user's knowledge, style, or methodology of operating the device in circumstances. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device are stored on a remote computing device or a remote computing system. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device include an artificial intelligence system for knowledge structuring, storing, or representation. The artificial intelligence system for knowledge structuring, storing, or representation may include at least one of: a deep learning system, a supervised learning system, an unsupervised learning system, a neural network, a search-based system, an optimization-based system, a logic-based system, a fuzzy logic-based system, a tree-based system, a graph-based system, a hierarchical system, a symbolic system, a sub-symbolic system, an evolutionary system, a genetic system, a multi-agent system, a deterministic system, a probabilistic system, or a statistical system.

In certain embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes comparing at least one portion of the new stream of collections of object representations with at least one portion of the first stream of collections of object representations. In further embodiments, the at least one portion of the new stream of collections of object representations include at least one collection of object representations, at least one object representation, or at least one object property of the new stream of collec-

tions of object representations. In further embodiments, the at least one portion of the first stream of collections of object representations include at least one collection of object representations, at least one object representation, or at least one object property of the first stream of collections of object representations. In further embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes comparing at least one collection of object representations from the new stream of collections of object representations with at least one collection of object representations from the first stream of collections of object representations. In further embodiments, the comparing at least one collection of object representations from the new stream of collections of object representations with at least one collection of object representations from the first stream of collections of object representations includes comparing at least one object representation of the at least one collection of object representations from the new stream of collections of object representations with at least one object representation of the at least one collection of object representations from the first stream of collections of object representations. The comparing at least one object representation of the at least one collection of object representations from the new stream of collections of object representations with at least one object representation of the at least one collection of object representations from the first stream of collections of object representations may include comparing at least one object property of the at least one object representation of the at least one collection of object representations from the new stream of collections of object representations with at least one object property of the at least one object representation of the at least one collection of object representations from the first stream of collections of object representations.

In some embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that there is at least a partial match between one or more portions of the new stream of collections of object representations and one or more portions of the first stream of collections of object representations. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that a similarity between at least one portion of the new stream of collections of object representations and at least one portion of the first stream of collections of object representations exceeds a similarity threshold. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining a substantial similarity between at least one portion of the new stream of collections of object representations and at least one portion of the first stream of collections of object representations. The substantial similarity may be achieved when a similarity between the at least one portion of the new stream of collections of object representations and the at least one portion of the first stream of collections of object representations exceeds a similarity threshold. The substantial similarity may be achieved when a number or a percentage of matching or partially matching portions of the new stream of collections of object representations and portions of the first stream of collections of object representations exceeds a threshold number or threshold percentage. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that a number or a percentage of matching or partially matching collections of object representations from the new stream of collections of object representations and from the first stream of collections of object representations exceeds a threshold number or threshold percentage. The matching or partially matching collections of object representations from the new stream of collections of object representations and from the first stream of collections of object representations may be determined factoring in at least one of: an importance of a collection of object representations, an order of a collection of object representations, a threshold for a similarity in a collection of object representations, or a threshold for a difference in a collection of object representations. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that a number or a percentage of matching or partially matching object representations from the new stream of collections of object representations and from the first stream of collections of object representations exceeds a threshold number or threshold percentage. The matching or partially matching object representations from the new stream of collections of object representations and from the first stream of collections of object representations may be determined factoring in at least one of: an association of an object representation with a collection of object representations, a type of an object representation, an importance of an object representation, a threshold for a similarity in an object representation, or a threshold for a difference in an object representation. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that a number or a percentage of matching or partially matching object properties from the new stream of collections of object representations and from the first stream of collections of object representations exceeds a threshold number or threshold percentage. The matching or partially matching object properties from the new stream of collections of object representations and from the first stream of collections of object representations may be determined factoring in at least one of: an association of an object property with an object representation, an association of an object property with a collection of object representations, a category of an object property, an importance of an object property, a threshold for a similarity in an object property, or a threshold for a difference in an object property. In further embodiments, determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that there is at least a partial match between at least one collection of object

representations from the new stream of collections of object representations and at least one collection of object representations from the first stream of collections of object representations. The determining that there is at least a partial match between at least one collection of object representations from the new stream of collections of object representations and at least one collection of object representations from the first stream of collections of object representations may include determining that there is at least a partial match between at least one object representation of the at least one collection of object representations from the new stream of collections of object representations and at least one object representation of the at least one collection of object representations from the first stream of collections of object representations. The determining that there is at least a partial match between at least one object representation of the at least one collection of object representations from the new stream of collections of object representations and at least one object representation of the at least one collection of object representations from the first stream of collections of object representations may include determining that there is at least a partial match between at least one object property of the at least one object representation of the at least one collection of object representations from the new stream of collections of object representations and at least one object property of the at least one object representation of the at least one collection of object representations from the first stream of collections of object representations.

In some embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations instead of or prior to an instruction set that would have been executed next. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations into a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting the processor circuit to the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting the processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes transmitting, to the processor circuit for execution, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes issuing an interrupt to the processor circuit and executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations following the interrupt. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying an element that is part of, operating on, or coupled to the processor circuit.

In certain embodiments, the processor circuit includes a logic circuit, and wherein the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the logic circuit includes a microcontroller. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying an element of the logic circuit. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations into an element of the logic circuit. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting the logic circuit to the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes replacing inputs into the logic circuit with the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes replacing outputs from the logic circuit with the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In some embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes causing an application for operating the device to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the application running on the processor circuit.

In certain embodiments, the system further comprises: an application including instruction sets for operating the device, the application running on the processor circuit, wherein the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying the application.

In some embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting an application to the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting an application to one or more alternate instruction sets, the application running on the processor circuit, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying at least one of: the memory unit, a register of the processor circuit, a storage, or a repository where instruction sets are stored or used. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets for operating an application or an object of the application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying at least one of: an element of the processor circuit, an element of the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing an assembly language. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes adding or inserting additional code into a code of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes at least one of: modifying, removing, rewriting, or overwriting a code of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of an application, the application running on the processor circuit. The branching or redirecting the code may include inserting at least one of: a branch, a jump, or a means for redirecting an execution. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of

object representations includes implementing a user's knowledge, style, or methodology of operating the device in a circumstance.

In certain embodiments, the system further comprises: an interface configured to cause execution of instruction sets, wherein the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations is caused by the interface. The interface may include a modification interface.

In some embodiments, the performing the one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes implementing a user's knowledge, style, or methodology of operating the device in a circumstance.

In certain embodiments, the system further comprises: an application running on the processor circuit.

In some embodiments, the instruction sets for operating the device are part of an application for operating the device, the application running on the processor circuit.

In certain embodiments, the system further comprises: an application for operating the device, the application running on the processor circuit. The application for operating the device may include the instruction sets for operating the device.

In some embodiments, the artificial intelligence unit is further configured to: receive at least one extra information. In further embodiments, the at least one extra information include one or more of: a time information, a location information, a computed information, or a contextual information. In further embodiments, the at least one extra information include one or more of: an information on an object, an information on an object representation, an information on a collection of object representations, an information on a stream of collections of object representations, an information on a device's circumstance, an information on an instruction set, an information on an application, an information on the processor circuit, an information on the device, or an information on an user. In further embodiments, the artificial intelligence unit is further configured to: learn the first stream of collections of object representations correlated with the at least one extra information. The learning the first stream of collections of object representations correlated with at least one extra information may include correlating the first stream of collections of object representations with the at least one extra information. The learning the first stream of collections of object representations correlated with at least one extra information may include storing the first stream of collections of object representations correlated with the at least one extra information into the memory unit. In further embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations. The anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations may include comparing an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations. The anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations may include determining that a similarity between an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations exceeds a similarity threshold.

In certain embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: cause the user interface to present a user with an option to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In some embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: receive, via the user interface, a user's selection to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In certain embodiments, the artificial intelligence unit is further configured to: rate the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. The rating the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations may include causing a user interface to display the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations along with one or more rating values as options to be selected by a user. The rating the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations may include rating the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations without a user input.

In some embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: cause the user interface to present a user with an option to cancel the execution of the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the canceling the execution of the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes restoring the processor circuit or the device to a prior state. The restoring the processor circuit or the device to a prior state may include saving the state of the processor circuit or the device prior to executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In certain embodiments, the system further comprises: an input device configured to receive a user's operating directions, the user's operating directions for instructing the processor circuit on how to operate the device.

In some embodiments, the autonomous device operating includes a partially or a fully autonomous device operating. The partially autonomous device operating may include executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations responsive to a user confirmation. The fully autonomous device operating may include executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations without a user confirmation.

In certain embodiments, the artificial intelligence unit is further configured to: receive a second stream of collections of object representations, the second stream of collections of object representations including one or more collections of representations of objects detected by the sensor; receive a second one or more instruction sets for operating the device; and learn the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device. In further embodiments, the second stream of collections of object representations includes one or more collections of representations of objects detected by the sensor over time. In further embodiments, the second stream of collections of object representations includes or is associated with a time stamp, an order, or a time related information. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device include creating a connection between the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device. The connection may include or is associated with at least one of: an occurrence count, a weight, a parameter, or a data. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device include updating a connection between the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device. In further embodiments, the updating the connection between the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device includes updating at least one of: an occurrence count, a weight, a parameter, or a data included in or associated with the connection. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device includes storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device into a first node of a knowledgebase, and wherein the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device includes storing the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device into a second node of the knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. In further embodiments, the knowledgebase may be stored in the memory unit. The learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device may include creating a connection between the first node and the second node. The learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device may include updating a connection between the first node and the second node. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device is stored into a first node of a neural network and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device is stored into a second node of the neural network. The first node and the second node may be connected by a connection. The first node may be part of a first layer of the neural network and the second node may be part of a second layer of the neural network. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device is stored into a first node of a graph and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device is stored into a second node of the graph. The first node and the second node may be connected by a connection. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device is stored into a first node of a sequence and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device is stored into a second node of the sequence.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first stream of collections of object representations, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The operations may further comprise: receiving a first one or more instruction sets for operating a device. The operations may further comprise: learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device. The operations may further comprise: receiving a new stream of collections of object representations, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The operations may further comprise: anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of

object representations. The operations may further comprise: causing an execution of the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the execution.

In certain embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from the one or more processor circuits or from another one or more processor circuits. In further embodiments, the execution of the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations is performed by the one or more processor circuits or by another one or more processor circuits.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first stream of collections of object representations by a processor circuit, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more instruction sets for operating a device by the processor circuit. The method may further comprise: (c) learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the processor circuit. The method may further comprise: (d) receiving a new stream of collections of object representations by the processor circuit, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The method may further comprise: (e) anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, the anticipating of (e) performed by the processor circuit. The method may further comprise: (f) executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the executing of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the executing of (f).

In some embodiments, the receiving of (b) includes receiving the first one or more instruction sets for operating the device from the processor circuit or from another processor circuit. In further embodiments, the executing of (f) is performed by the processor circuit or by another processor circuit.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable as well as the following embodiments.

In certain embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device includes storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device into a memory unit, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device being part of a plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device stored in the memory unit.

In some embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations instead of or prior to an instruction set that would have been executed next. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying a register or an element of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations into a register or an element of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting a processor circuit to the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting a processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes transmitting, to a processor circuit for execution, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes issuing an interrupt to a processor circuit and executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations following the interrupt. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of

object representations includes modifying an element that is part of, operating on, or coupled to a processor circuit.

In certain embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes executing, by a logic circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the logic circuit includes a microcontroller. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying an element of the logic circuit. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations into an element of the logic circuit. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting the logic circuit to the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes replacing inputs into the logic circuit with the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes replacing outputs from the logic circuit with the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In some embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes executing, by an application for operating the device, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying an application, the application including instruction sets for operating the device. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting an application to the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting an application to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying at least one of: the memory unit, a register of a processor circuit, a storage, or a repository where instruction sets are stored or used. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets for operating an application or an object of the application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying at least one of: an element of a processor circuit, an element of the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing an assembly language. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation,

JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes adding or inserting additional code into a code of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes at least one of: modifying, removing, rewriting, or overwriting a code of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of an application. The branching or redirecting the code may include inserting at least one of: a branch, a jump, or a means for redirecting an execution. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes implementing a user's knowledge, style, or methodology of operating the device in a circumstance. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations via an interface. The interface may include a modification interface.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving at least one extra information. In further embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: learning the first stream of collections of object representations correlated with the at least one extra information.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: presenting, via a user interface, a user with an option to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving, via a user interface, a user's selection to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: rating the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: presenting, via a user interface, a user with an option to cancel the execution of the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving, via an input device, a user's operating directions, the user's operating directions for instructing a processor circuit on how to operate the device.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving a second stream of collections of object representations, the second stream of collections of object representations including one or more collections of representations of objects detected by the sensor; receiving a second one or more instruction sets for operating the device; and learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device. In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving, by a first processor circuit of the one or more processor circuits, a first stream of collections of object representations, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The operations may further comprise: receiving, by the first processor circuit of the one or more processor circuits, a first one or more instruction sets for operating a device. The operations may further comprise: learning, by the first processor circuit of the one or more processor circuits, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device. The operations may further comprise: receiving, by the first processor circuit of the one or more processor circuits, a new stream of collections of object representations, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The operations may further comprise: anticipating, by the first processor circuit of the one or more processor circuits, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The operations may further comprise: causing, by the first processor circuit of the one or more processor circuits, an execution, by a second processor circuit of the one or more processor circuits, of the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the causing performed in response to the anticipating, by the first processor circuit of the one or more processor circuits, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the execution.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first stream of collections of object representations by a first processor circuit, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more instruction sets for operating a device by the

first processor circuit. The method may further comprise: (c) learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the first processor circuit. The method may further comprise: (d) receiving a new stream of collections of object representations by the first processor circuit, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The method may further comprise: (e) anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, the anticipating of (e) performed by the first processor circuit. The method may further comprise: (f) executing, by a second processor circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the executing of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the executing of (f).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for learning a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first stream of collections of object representations, the first stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: learn the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first stream of collections of object representations, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The operations may further comprise: receiving a first one or more instruction sets for operating a device. The operations may further comprise: learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a non method comprising: (a) receiving a first stream of collections of object representations by a processor circuit, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more instruction sets for operating a device by the processor circuit. The method may further comprise: (c) learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the processor circuit.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: access the memory unit that comprises a plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device including a first stream of collections of object representations correlated with a first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: receive a new stream of collections of object representations, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations performed in response to the executing by the processor circuit.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising:

accessing a memory unit that comprises a plurality of streams of collections of object representations correlated with one or more instruction sets for operating a device, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device including a first stream of collections of object representations correlated with a first one or more instruction sets for operating the device. The operations may further comprise: receiving a new stream of collections of object representations, the new stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The operations may further comprise: anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The operations may further comprise: causing an execution of the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the execution.

In some aspects, the disclosure relates to a method comprising: (a) accessing a memory unit that comprises a plurality of streams of collections of object representations correlated with one or more instruction sets for operating a device, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device including a first stream of collections of object representations correlated with a first one or more instruction sets for operating the device, the accessing of (a) performed by a processor circuit. The method may further comprise: (b) receiving a new stream of collections of object representations by the processor circuit, the new stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The method may further comprise: (c) anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, the anticipating of (c) performed by the processor circuit. The method may further comprise: (d) executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the executing of (d) performed in response to the anticipating of (c). The method may further comprise: (e) performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the executing of (d).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for learning and using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a logic circuit configured to receive inputs and produce outputs, wherein the outputs are used for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first collection of object representations, the first collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more inputs, wherein the first one or more inputs are also received by the logic circuit. The artificial intelligence unit may be further configured to: learn the first collection of object representations correlated with the first one or more inputs. The artificial intelligence unit may be further configured to: receive a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may be further configured to: cause the logic circuit to receive the first one or more inputs correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by one or more outputs for operating the device produced by the logic circuit.

In some embodiments, the logic circuit configured to receive inputs and produce outputs includes a logic circuit configured to produce outputs based at least in part on logic operations performed on the inputs. In further embodiments, the learning the first collection of object representations correlated with the first one or more inputs includes correlating the first collection of object representations with the first one or more inputs. In further embodiments, the learning the first collection of object representations correlated with the first one or more inputs includes storing the first collection of object representations correlated with the first one or more inputs into the memory unit, the first collection of object representations correlated with the first one or more inputs being part of a plurality of collections of object representations correlated with one or more inputs stored in the memory unit. In further embodiments, the anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes comparing at least one portion of the new collection of object representations with at least one portion of the first collection of object representations. In further embodiments, the anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations

includes determining that there is at least a partial match between the new collection of object representations and the first collection of object representations. In further embodiments, the causing the logic circuit to receive the first one or more inputs correlated with the first collection of object representations includes transmitting, to the logic circuit, the first one or more inputs correlated with the first collection of object representations. In further embodiments, the causing the logic circuit to receive the first one or more inputs correlated with the first collection of object representations includes replacing one or more inputs into the logic circuit with the first one or more inputs correlated with the first collection of object representations.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first collection of object representations, the first collection of object representations including one or more representations of objects detected by a sensor. The operations may further comprise: receiving a first one or more inputs, wherein the first one or more inputs are also received by a logic circuit, and wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device. The operations may further comprise: learning the first collection of object representations correlated with the first one or more inputs. The operations may further comprise: receiving a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The operations may further comprise: anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further comprise: causing the logic circuit to receive the first one or more inputs correlated with the first collection of object representations, the causing performed in response to the anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, wherein the device performs one or more operations defined by one or more outputs for operating the device produced by the logic circuit.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a processor circuit, the first collection of object representations including one or more representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more inputs by the processor circuit, wherein the first one or more inputs are also received by a logic circuit, and wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device. The method may further comprise: (c) learning the first collection of object representations correlated with the first one or more inputs, the learning of (c) performed by the processor circuit. The method may further comprise: (d) receiving a new collection of object representations by the processor circuit, the new collection of object representations including one or more representations of objects detected by the sensor. The method may further comprise: (e) anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (e) performed by the processor circuit. The method may further comprise: (f) receiving, by the logic circuit, the first one or more inputs correlated with the first collection of object representations, the receiving of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the device, one or more operations defined by one or more outputs for operating the device produced by the logic circuit.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for learning and using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a logic circuit configured to receive inputs and produce outputs, wherein the outputs are used for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence. The artificial intelligence unit may be configured to: receive a first collection of object representations, the first collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more outputs, the first one or more outputs transmitted from the logic circuit. The artificial intelligence unit may be further configured to: learn the first collection of object representations correlated with the first one or more outputs. The artificial intelligence unit may be further configured to: receive a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more outputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may be further configured to: cause the device to perform one or more operations defined by the first one or more outputs correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit.

In some embodiments, the learning the first collection of object representations correlated with the first one or more outputs includes correlating the first collection of object representations with the first one or more outputs. In further embodiments, the learning the first collection of object representations correlated with the first one or more outputs includes storing the first collection of object representations correlated with the first one or more outputs into the memory unit, the first collection of object representations correlated with the first one or more outputs being part of a plurality of collections of object representations correlated with one or more outputs stored in the memory unit. In further embodiments, the anticipating the first one or more outputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes comparing at least one portion of

the new collection of object representations with at least one portion of the first collection of object representations. In further embodiments, the anticipating the first one or more outputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes determining that there is at least a partial match between the new collection of object representations and the first collection of object representations. In further embodiments, the causing the device to perform one or more operations defined by the first one or more outputs correlated with the first collection of object representations includes replacing one or more outputs from the logic circuit with the first one or more outputs correlated with the first collection of object representations.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first collection of object representations, the first collection of object representations including one or more representations of objects detected by a sensor. The operations may further comprise: receiving a first one or more outputs, the first one or more outputs transmitted from a logic circuit, wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device. The operations may further comprise: learning the first collection of object representations correlated with the first one or more outputs. The operations may further comprise: receiving a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The operations may further comprise: anticipating the first one or more outputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further comprise: causing the device to perform one or more operations defined by the first one or more outputs correlated with the first collection of object representations, the causing performed in response to the anticipating the first one or more outputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a processor circuit, the first collection of object representations including one or more representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more outputs by the processor circuit, the first one or more outputs transmitted from a logic circuit, wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device. The method may further comprise: (c) learning the first collection of object representations correlated with the first one or more outputs, the learning of (c) performed by the processor circuit. The method may further comprise: (d) receiving a new collection of object representations by the processor circuit, the new collection of object representations including one or more representations of objects detected by the sensor. The method may further comprise: (e) anticipating the first one or more outputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (e) performed by the processor circuit. The method may further comprise: (f) performing, by the device, one or more operations defined by the first one or more outputs correlated with the first collection of object representations, the one or more operations by the device performed in response to the anticipating of (e).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for learning and using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: an actuator configured to receive inputs and perform motions. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first collection of object representations, the first collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more inputs, wherein the first one or more inputs are also received by the actuator. The artificial intelligence unit may be further configured to: learn the first collection of object representations correlated with the first one or more inputs. The artificial intelligence unit may be further configured to: receive a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may be further configured to: cause the actuator to receive the first one or more inputs correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the actuator performs one or more motions defined by the first one or more inputs correlated with the first collection of object representations.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first collection of object representations, the first collection of object representations including one or more representations of objects detected by a sensor. The operations may further comprise: receiving a first one or more inputs, wherein the first one or more inputs are also received by an actuator, and wherein the actuator is configured to receive inputs and perform motions. The operations may further comprise: learning the first collection of object representations correlated with the first one or more inputs. The operations may further comprise: receiving a new collection of object representations, the new collection of object representations including one or more representations

of objects detected by the sensor. The operations may further comprise: anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further comprise: causing the actuator to receive the first one or more inputs correlated with the first collection of object representations, the causing performed in response to the anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, wherein the actuator performs one or more motions defined by the first one or more inputs correlated with the first collection of object representations.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a processor circuit, the first collection of object representations including one or more representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more inputs by the processor circuit, wherein the first one or more inputs are also received by an actuator, and wherein the actuator is configured to receive inputs and perform motions. The method may further comprise: (c) learning the first collection of object representations correlated with the first one or more inputs, the learning of (c) performed by the processor circuit. The method may further comprise: (d) receiving a new collection of object representations by the processor circuit, the new collection of object representations including one or more representations of objects detected by the sensor. The method may further comprise: (e) anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (e) performed by the processor circuit. The method may further comprise: (f) receiving, by the actuator, the first one or more inputs correlated with the first collection of object representations, the receiving of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the actuator, one or more motions defined by the first one or more inputs correlated with the first collection of object representations.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

Other features and advantages of the disclosure will become apparent from the following description, including the claims and drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** illustrates a block diagram of Computing Device **70** that can provide processing capabilities used in some of the disclosed embodiments.

FIG. **2** illustrates an embodiment of Device **98** comprising Unit for Learning and/or Using a Device's Circumstances for Autonomous Device Operation (DCADO Unit **100**).

FIGS. **3**A-**3**E illustrate various embodiments of Sensors **92** and elements of Object Processing Unit **93**.

FIGS. **4**A-**4**B, illustrate an exemplary embodiment of Objects **615** detected in Device's **98** surrounding, and resulting Collection of Object Representations **525**.

FIG. **5** illustrates some embodiments of obtaining instruction sets, data, and/or other information through tracing, profiling, or sampling of Processor **11** registers, memory, or other computing system components.

FIGS. **6**A-**6**B illustrate some embodiments of obtaining instruction sets, data, and/or other information through tracing, profiling, or sampling of Logic Circuit **250**.

FIGS. **7**A-**7**E illustrate some embodiments of Instruction Sets **526**.

FIGS. **8**A-**8**B illustrate some embodiments of Extra Information **527**.

FIG. **9** illustrates an embodiment where DCADO Unit **100** is part of or operating on Processor **11**.

FIG. **10** illustrates an embodiment where DCADO Unit **100** resides on Server **96** accessible over Network **95**.

FIG. **11** illustrates an embodiment of learning and/or using Remote Device's **97** circumstances for autonomous Remote Device **97** operation.

FIG. **12** illustrates an embodiment of Artificial Intelligence Unit **110**.

FIG. **13** illustrates an embodiment of Knowledge Structuring Unit **520** correlating individual Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527**.

FIG. **14** illustrates another embodiment of Knowledge Structuring Unit **520** correlating individual Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527**.

FIG. **15** illustrates an embodiment of Knowledge Structuring Unit **520** correlating streams of Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527**.

FIG. **16** illustrates another embodiment of Knowledge Structuring Unit **520** correlating streams of Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527**.

FIG. **17** illustrates various artificial intelligence methods, systems, and/or models that can be utilized in DCADO Unit **100** embodiments.

FIG. **18**A-**18**C illustrate embodiments of interconnected Knowledge Cells **800** and updating weights of Connections **853**.

FIG. **19** illustrates an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Collection of Knowledge Cells **530***d*.

FIG. **20** illustrates an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Neural Network **530***a*.

FIG. **21** illustrates an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Neural Network **530***a* comprising shortcut Connections **853**.

FIG. **22** illustrates an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Graph **530***b*.

FIG. **23** illustrates an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Collection of Sequences **530***c*.

FIG. **24** illustrates an embodiment of determining anticipatory Instruction Sets **526** from a single Knowledge Cell **800**.

FIG. **25** illustrates an embodiment of determining anticipatory Instruction Sets **526** by traversing a single Knowledge Cell **800**.

FIG. **26** illustrates an embodiment of determining anticipatory Instruction Sets **526** using collective similarity comparisons.

FIG. **27** illustrates an embodiment of determining anticipatory Instruction Sets **526** using Neural Network **530***a*.

FIG. **28** illustrates an embodiment of determining anticipatory Instruction Sets **526** using Graph **530***b*.

FIG. **29** illustrates an embodiment of determining anticipatory Instruction Sets **526** using Collection of Sequences **530***c*.

FIG. **30** illustrates some embodiments of modifying execution and/or functionality of Processor **11** through modification of Processor **11** registers, memory, or other computing system components.

FIGS. **31**A-**31**B illustrate some embodiments of modifying execution and/or functionality of Logic Circuit **250** through modification of inputs and/or outputs of Logic Circuit **250**.

FIG. **32** illustrates a flow chart diagram of an embodiment of method **9100** for learning and/or using a device's circumstances for autonomous device operation.

FIG. **33** illustrates a flow chart diagram of an embodiment of method **9200** for learning and/or using a device's circumstances for autonomous device operation.

FIG. **34** illustrates a flow chart diagram of an embodiment of method **9300** for learning and/or using a device's circumstances for autonomous device operation.

FIG. **35** illustrates a flow chart diagram of an embodiment of method **9400** for learning and/or using a device's circumstances for autonomous device operation.

FIG. **36** illustrates a flow chart diagram of an embodiment of method **9500** for learning and/or using a device's circumstances for autonomous device operation.

FIG. **37** illustrates a flow chart diagram of an embodiment of method **9600** for learning and/or using a device's circumstances for autonomous device operation.

FIG. **38** illustrates an exemplary embodiment of Loader **98***a*.

FIG. **39** illustrates an exemplary embodiment of Boat **98***b*.

FIG. **40** illustrates an exemplary embodiment of utilizing Area of Interest **450** around Boat **98***b*.

Like reference numerals in different figures indicate like elements. Horizontal or vertical " . . . " or other such indicia may be used to indicate additional instances of the same type of element n, m, x, or other such letters or indicia represent integers or other sequential numbers that follow the sequence where they are indicated. It should be noted that n, m, x, or other such letters or indicia may represent different numbers in different elements even where the elements are depicted in the same figure. In general, n, m, x, or other such letters or indicia may follow the sequence and/or context where they are indicated. Any of these or other such letters or indicia may be used interchangeably depending on context and space available. The drawings are not necessarily to scale, with emphasis instead being placed upon illustrating the embodiments, principles, and concepts of the disclosure. A line or arrow between any of the disclosed elements comprises an interface that enables the coupling, connection, and/or interaction between the elements.

DETAILED DESCRIPTION

The disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation comprise apparatuses, systems, methods, features, functionalities, and/or applications that enable learning a device's circumstances including objects with various properties along with correlated instruction sets for operating the device, storing this knowledge in a knowledgebase (i.e. neural network, graph, sequences, etc.), and operating a device autonomously. The disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, any of their elements, any of their embodiments, or a combination thereof can generally be referred to as DCADO, DCADO Unit, or as other suitable name or reference.

Referring now to FIG. **1**, an embodiment is illustrated of Computing Device **70** (also referred to simply as computing device, computing system, or other suitable name or reference, etc.) that can provide processing capabilities used in some embodiments of the forthcoming disclosure. Later described devices, systems, and methods, in combination with processing capabilities of Computing Device **70**, enable learning and/or using a device's circumstances for autonomous device operation and/or other functionalities described herein. Various embodiments of the disclosed devices, systems, and methods include hardware, functions, logic, programs, and/or a combination thereof that can be implemented using any type or form of computing, computing enabled, or other device or system such as a mobile device, a computer, a computing enabled telephone, a server, a gaming device, a television device, a digital camera, a GPS receiver, a media player, an embedded device, a supercomputer, a wearable device, an implantable device, a cloud, or any other type or form of computing, computing enabled, or other device or system capable of performing the operations described herein.

In some designs, Computing Device **70** comprises hardware, processing techniques or capabilities, programs, or a combination thereof. Computing Device **70** includes one or more central processing units, which may also be referred to as processors **11**. Processor **11** includes one or more memory ports **10** and/or one or more input-output ports, also referred to as I/O ports **15**, such as I/O ports **15**A and **15**B. Processor **11** may be special or general purpose. Computing Device **70** may further include memory **12**, which can be connected to the remainder of the components of Computing Device **70** via bus **5**. Memory **12** can be connected to processor **11** via memory port **10**. Computing Device **70** may also include display device **21** such as a monitor, projector, glasses, and/or other display device. Computing Device **70** may also include Human-machine Interface **23** such as a keyboard, a pointing device, a mouse, a touchscreen, a joystick, a remote controller, and/or other input device. In some implementations, Human-machine Interface **23** can be connected with bus **5** or directly connected with specific elements of Computing Device **70**. Computing Device **70** may include additional elements such as one or more input/output devices **13**. Processor **11** may include or be interfaced with cache memory **14**. Storage **27** may include memory, which provides an operating system **17** (i.e. also referred to as OS **17**, etc.), additional application programs **18**, and/or data space **19** in which additional data or information can be stored. Alternative memory device **16** can be connected to the remaining components of Computing Device **70** via bus **5**. Network interface **25** can also be connected with bus **5** and be used to communicate with external computing devices via a network. Some or all described elements of Computing Device **70** can be directly or operatively connected or coupled with each other using any other connection means

known in art. Other additional elements may be included as needed, or some of the disclosed ones may be excluded, or a combination thereof may be utilized in alternate implementations of Computing Device **70**.

Processor **11** includes one or more circuits or devices that can execute instructions fetched from memory **12** and/or other element. Processor **11** may include any combination of hardware and/or processing techniques or capabilities for executing or implementing logic functions or programs. Processor **11** may include a single core or a multi core processor. Processor **11** includes the functionality for loading operating system **17** and operating any application programs **18** thereon. In some embodiments, Processor **11** can be provided in a microprocessing or a processing unit, such as, for example, Snapdragon processor produced by Qualcomm Inc., processor by Intel Corporation of Mountain View, Calif., processor manufactured by Motorola Corporation of Schaumburg, Ill.; processor manufactured by Transmeta Corporation of Santa Clara, Calif.; processor manufactured by International Business Machines of White Plains, N.Y.; processor manufactured by Advanced Micro Devices of Sunnyvale, Calif., or any computing circuit or device for performing similar functions. In other embodiments, processor **11** can be provided in a graphics processing unit (GPU), visual processing unit (VPU), or other highly parallel processing circuit or device such as, for example, nVidia GeForce line of GPUs, AMD Radeon line of GPUs, and/or others. Such GPUs or other highly parallel processing circuits or devices may provide superior performance in processing operations on neural networks, graphs, and/or other data structures. In further embodiments, processor **11** can be provided in a micro controller such as, for example, Texas instruments, Atmel, Microchip Technology, ARM, Silicon Labs, Intel, and/or other lines of micro controllers. In further embodiments, processor **11** can be provided in a quantum processor such as, for example, D-Wave Systems, Microsoft, Intel, IBM, Google, Toshiba, and/or other lines of quantum processors. In further embodiments, processor **11** can be provided in a biocomputer such as DNA-based computer, protein-based computer, molecule-based computer, and/or others. In further embodiments, processor **11** includes any circuit or device for performing logic operations. Processor **11** can be based on any of the aforementioned or other available processors capable of operating as described herein. Computing Device **70** may include one or more of the aforementioned or other processors. In some designs, processor **11** can communicate with memory **12** via a system bus **5**. In other designs, processor **11** can communicate directly with memory **12** via a memory port **10**.

Memory **12** includes one or more circuits or devices capable of storing data. In some embodiments, Memory **12** can be provided in a semiconductor or electronic memory chip such as static random access memory (SRAM), Flash memory, Burst SRAM or SynchBurst SRAM (BSRAM), Dynamic random access memory (DRAM), Fast Page Mode DRAM (FPM DRAM), Enhanced DRAM (EDRAM), Extended Data Output RAM (EDO RAM), Extended Data Output DRAM (EDO DRAM), Burst Extended Data Output DRAM (BEDO DRAM), Enhanced DRAM (EDRAM), synchronous DRAM (SDRAM), JEDEC SRAM, PC100 SDRAM, Double Data Rate SDRAM (DDR SDRAM), Enhanced SDRAM (ESDRAM), SyncLink DRAM (SLDRAM), Direct Rambus DRAM (DRDRAM), Ferroelectric RAM (FRAM), and/or others. In other embodiments, Memory **12** includes any volatile memory. In general, Memory **12** can be based on any of the aforementioned or other available memories capable of operating as described herein.

Storage **27** includes one or more devices or mediums capable of storing data. In some embodiments, Storage **27** can be provided in a device or medium such as a hard drive, flash drive, optical disk, and/or others. In other embodiments, Storage **27** can be provided in a biological storage device such as DNA-based storage device, protein-based storage device, molecule-based storage device, and/or others. In further embodiments, Storage **27** can be provided in an optical storage device such as holographic storage, and/or others. In further embodiments, Storage **27** may include any non-volatile memory. In general, Storage **27** can be based on any of the aforementioned or other available storage devices or mediums capable of operating as described herein. In some aspects, Storage **27** may include any features, functionalities, and embodiments of Memory **12**, and vice versa, as applicable.

Processor **11** can communicate directly with cache memory **14** via a connection means such as a secondary bus which may also sometimes be referred to as a backside bus. In some embodiments, processor **11** can communicate with cache memory **14** using the system bus **5**. Cache memory **14** may typically have a faster response time than main memory **12** and can include a type of memory which is considered faster than main memory **12** such as, for example, SRAM, BSRAM, or EDRAM. Cache memory includes any structure such as multilevel caches, for example. In some embodiments, processor **11** can communicate with one or more I/O devices **13** via a system bus **5**. Various busses can be used to connect processor **11** to any of the I/O devices **13** such as a VESA VL bus, an ISA bus, an EISA bus, a MicroChannel Architecture (MCA) bus, a PCI bus, a PCI-X bus, a PCI-Express bus, a NuBus, and/or others. In some embodiments, processor **11** can communicate directly with I/O device **13** via HyperTransport, Rapid I/O, or InfiniBand. In further embodiments, local busses and direct communication can be mixed. For example, processor **11** can communicate with an I/O device **13** using a local interconnect bus and communicate with another I/O device **13** directly. Similar configurations can be used for any other components described herein.

Computing Device **70** may further include alternative memory such as a SD memory slot, a USB memory stick, an optical drive such as a CD-ROM drive, a CD-R/RW drive, a DVD-ROM drive or a BlueRay disc, a hard-drive, and/or any other device comprising non-volatile memory suitable for storing data or installing application programs. Computing Device **70** may further include a storage device **27** comprising any type or form of non-volatile memory for storing an operating system (OS) such as any type or form of Windows OS, Mac OS, Unix OS, Linux OS, Android OS, iPhone OS, mobile version of Windows OS, an embedded OS, or any other OS that can operate on Computing Device **70**. Computing Device **70** may also include application programs **18**, and/or data space **19** for storing additional data or information. In some embodiments, alternative memory **16** can be used as or similar to storage device **27**. Additionally, OS **17** and/or application programs **18** can be operable from a bootable medium such as, for example, a flash drive, a micro SD card, a bootable CD or DVD, and/or other bootable medium.

Application Program **18** (also referred to as program, computer program, application, script, code, or other suitable name or reference) comprises instructions that can provide functionality when executed by processor **11**. As such, Application Program **18** may be used to operate (i.e.

perform operations on/with) or control a device or system. Application program **18** can be implemented in a high-level procedural or object-oriented programming language, or in a low-level machine or assembly language. Any language used can be compiled, interpreted, or otherwise translated into machine language. Application program **18** can be deployed in any form including as a stand-alone program or as a module, component, subroutine, or other unit suitable for use in a computing system. Application program **18** does not necessarily correspond to a file in a file system. A program can be stored in a portion of a file that may hold other programs or data, in a single file dedicated to the program, or in multiple files (i.e. files that store one or more modules, sub programs, or portions of code, etc.). Application Program **18** can be delivered in various forms such as, for example, executable file, library, script, plugin, addon, applet, interface, console application, web application, application service provider (ASP)-type application, operating system, and/or other forms. Application program **18** can be deployed to be executed on one computing device or on multiple computing devices (i.e. cloud, distributed, or parallel computing, etc.), or at one site or distributed across multiple sites interconnected by a communication network or an interface.

Network interface **25** can be utilized for interfacing Computing Device **70** with other devices via a network through a variety of connections including telephone lines, wired or wireless connections, LAN or WAN links (i.e. 802.11, T1, T3, 56 kb, X.25, etc.), broadband connections (i.e. ISDN, Frame Relay, ATM, etc.), or a combination thereof. Examples of networks include the Internet, an intranet, an extranet, a local area network (LAN), a wide area network (WAN), a personal area network (PAN), a home area network (HAN), a campus area network (CAN), a metropolitan area network (MAN), a global area network (GAN), a storage area network (SAN), virtual network, a virtual private network (VPN), a Bluetooth network, a wireless network, a wireless LAN, a radio network, a HomePNA, a power line communication network, a G.hn network, an optical fiber network, an Ethernet network, an active networking network, a client-server network, a peer-to-peer network, a bus network, a star network, a ring network, a mesh network, a star-bus network, a tree network, a hierarchical topology network, and/or other networks. Network interface **25** may include a built-in network adapter, network interface card, PCMCIA network card, card bus network adapter, wireless network adapter, Bluetooth network adapter, WiFi network adapter, USB network adapter, modem, and/or any other device suitable for interfacing Computing Device **70** with any type of network capable of communication and/or operations described herein.

I/O devices **13** may be present in various shapes or forms in Computing Device **70**. Examples of I/O device **13** capable of input include a joystick, a keyboard, a mouse, a trackpad, a trackpoint, a touchscreen, a trackball, a microphone, a drawing tablet, a glove, a tactile input device, a still or video camera, and/or other input device. Examples of I/O device **13** capable of output include a video display, a touchscreen, a projector, a glasses, a speaker, a tactile output device, and/or other output device. Examples of I/O device **13** capable of input and output include a disk drive, an optical storage device, a modem, a network card, and/or other input/output device. I/O device **13** can be interfaced with processor **11** via an I/O port **15**, for example. In some aspects, I/O device **13** can be a bridge between system bus **5** and an external communication bus such as a USB bus, an Apple Desktop Bus, an RS-232 serial connection, a SCSI bus, a FireWire bus, a FireWire 800 bus, an Ethernet bus, an AppleTalk bus, a Gigabit Ethernet bus, an Asynchronous Transfer Mode bus, a HIPPI bus, a Super HIPPI bus, a SerialPlus bus, a SCI/LAMP bus, a FibreChannel bus, a Serial Attached small computer system interface bus, and/or other bus.

An output interface (not shown) such as a graphical user interface, an acoustic output interface, a tactile output interface, any device driver (i.e. audio, video, or other driver), and/or other output interface or system can be utilized to process output from elements of Computing Device **70** for conveyance on an output device such as Display **21**. In some aspects, Display **21** or other output device itself may include an output interface for processing output from elements of Computing Device **70**. Further, an input interface (not shown) such as a keyboard listener, a touchscreen listener, a mouse listener, any device driver (i.e. audio, video, keyboard, mouse, touchscreen, or other driver), and/or other input interface or system can be utilized to process input from Human-machine Interface **23** or other input device for use by elements of Computing Device **70**. In some aspects, Human-machine Interface **23** or other input device itself may include an input interface for processing input for use by elements of Computing Device **70**.

Computing Device **70** may include or be connected to multiple display devices **21**. Display devices **21** can each be of the same or different type or form. Computing Device **70** and/or its elements comprise any type or form of suitable hardware, programs, or a combination thereof to support, enable, or provide for the connection and use of multiple display devices **21**. In one example, Computing Device **70** includes any type or form of video adapter, video card, driver, and/or library to interface, communicate, connect, or otherwise use display devices **21**. In some aspects, a video adapter may include multiple connectors to interface to multiple display devices **21**. In other aspects, Computing Device **70** includes multiple video adapters, with each video adapter connected to one or more display devices **21**. In some embodiments, Computing Device's **70** operating system can be configured for using multiple displays **21**. In other embodiments, one or more display devices **21** can be provided by one or more other computing devices such as remote computing devices connected to Computing Device **70** via a network or an interface.

Computing Device **70** can operate under the control of operating system **17**, which may support Computing Device's **70** basic functions, interface with and manage hardware resources, interface with and manage peripherals, provide common services for application programs, schedule tasks, and/or perform other functionalities. A modern operating system enables features and functionalities such as a high resolution display, graphical user interface (GUI), touchscreen, cellular network connectivity (i.e. mobile operating system, etc.), Bluetooth connectivity, WiFi connectivity, global positioning system (GPS) capabilities, mobile navigation, microphone, speaker, still picture camera, video camera, voice recorder, speech recognition, music player, video player, near field communication, personal digital assistant (PDA), and/or other features, functionalities, or applications. For example, Computing Device **70** can use any conventional operating system, any embedded operating system, any real-time operating system, any open source operating system, any video gaming operating system, any proprietary operating system, any online operating system, any operating system for mobile computing devices, or any other operating system capable of running on Computing

Device **70** and performing operations described herein. Example of operating systems include Windows XP, Windows 7, Windows 8, Windows 10, etc. manufactured by Microsoft Corporation of Redmond, Wash.; Mac OS, iPhone OS, etc. manufactured by Apple Computer of Cupertino, Calif.; OS/2 manufactured by International Business Machines of Armonk, N.Y.; Linux, a freely-available operating system distributed by Caldera Corp. of Salt Lake City, Utah; or any type or form of a Unix operating system, and/or others. Any operating systems such as the ones for Android devices can similarly be utilized.

Computing Device **70** can be implemented as or be part of various model architectures such as web services, distributed computing, grid computing, cloud computing, and/or other architectures. For example, in addition to the traditional desktop, server, or mobile operating system architectures, a cloud-based operating system can be utilized to provide the structure on which embodiments of the disclosure can be implemented. Other aspects of Computing Device **70** can also be implemented in the cloud without departing from the spirit and scope of the disclosure. For example, memory, storage, processing, and/or other elements can be hosted in the cloud. In some embodiments, Computing Device **70** can be implemented on multiple devices. For example, a portion of Computing Device **70** can be implemented on a mobile device and another portion can be implemented on wearable electronics.

Computing Device **70** can be or include any mobile device, a mobile phone, a smartphone (i.e. iPhone, Windows phone, Blackberry phone, Android phone, etc.), a tablet, a personal digital assistant (PDA), wearable electronics, implantable electronics, and/or other mobile device capable of implementing the functionalities described herein. Computing Device **70** can also be or include an embedded device, which can be any device or system with a dedicated function within another device or system. Embedded systems range from the simplest ones dedicated to one task with no user interface to complex ones with advanced user interface that may resemble modern desktop computer systems. Examples of devices comprising an embedded device include a mobile telephone, a personal digital assistant (PDA), a gaming device, a media player, a digital still or video camera, a pager, a television device, a set-top box, a personal navigation device, a global positioning system (GPS) receiver, a portable storage device (i.e. a USB flash drive, etc.), a digital watch, a DVD player, a printer, a microwave oven, a washing machine, a dishwasher, a gateway, a router, a hub, an automobile entertainment system, an automobile navigation system, a refrigerator, a washing machine, a factory automation device, an assembly line device, a factory floor monitoring device, a thermostat, an automobile, a factory controller, a telephone, a network bridge, and/or other devices. An embedded device can operate under the control of an operating system for embedded devices such as MicroC/OS-II, QNX, VxWorks, eCos, TinyOS, Windows Embedded, Embedded Linux, and/or other embedded device operating systems.

Various implementations of the disclosed devices, systems, and methods can be realized in digital electronic circuitry, integrated circuitry, logic gates, application specific integrated circuits (ASICs), field programmable gate arrays (FPGAs), computer hardware, firmware, programs, virtual machines, and/or combinations thereof including their structural, logical, and/or physical equivalents.

The disclosed devices, systems, and methods may include clients and servers. A client and server are generally, but not always, remote from each other and typically, but not always, interact via a network or an interface. The relationship of a client and server may arise by virtue of computer programs running on their respective computers and having a client-server relationship to each other, for example.

The disclosed devices, systems, and methods can be implemented in a computing system that includes a back end component, a middleware component, a front end component, or any combination thereof. The components of the system can be interconnected by any form or medium of digital data communication such as, for example, a network.

Computing Device **70** may include or be interfaced with a computer program product comprising instructions or logic encoded on a computer-readable medium. Such instructions or logic, when executed, may configure or cause one or more processors to perform the operations and/or functionalities disclosed herein. For example, a computer program can be provided or encoded on a computer-readable medium such as an optical medium (i.e. DVD-ROM, etc.), flash drive, hard drive, any memory, firmware, or other medium. Computer program can be installed onto a computing device to cause the computing device to perform the operations and/or functionalities disclosed herein. Machine-readable medium, computer-readable medium, or other such terms may refer to any computer program product, apparatus, and/or device for providing instructions and/or data to one or more programmable processors. As such, machine-readable medium includes any medium that can send and/or receive machine instructions as a machine-readable signal. Examples of a machine-readable medium include a volatile and/or non-volatile medium, a removable and/or non-removable medium, a communication medium, a storage medium, and/or other medium. A communication medium, for example, can transmit computer readable instructions and/or data in a modulated data signal such as a carrier wave or other transport technique, and may include any other form of information delivery medium known in art. A non-transitory machine-readable medium comprises all machine-readable media except for a transitory, propagating signal.

In some embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, can be implemented entirely or in part in a device (i.e. microchip, circuitry, logic gates, electronic device, computing device, special or general purpose processor, etc.) or system that comprises (i.e. hard coded, internally stored, etc.) or is provided with (i.e. externally stored, etc.) instructions for implementing DCADO functionalities. As such, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, may include the processing, memory, storage, and/or other features, functionalities, and embodiments of Computing Device **70** or elements thereof. Such device or system can operate on its own (i.e. standalone device or system, etc.), be embedded in another device or system (i.e. an industrial machine, a robot, a vehicle, a toy, a smartphone, a television device, an appliance, and/or any other device or system capable of housing the elements needed for DCADO functionalities), work in combination with other devices or systems, or be available in any other configuration. In other embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, may include Alternative Memory **16** that provides instructions for implementing DCADO functionalities to one or more Processors **11**. In further embodiments, the disclosed artificially intelligent

devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, can be implemented entirely or in part as a computer program and executed by one or more Processors **11**. Such program can be implemented in one or more modules or units of a single or multiple computer programs. Such program may be able to attach to or interface with, inspect, and/or take control of another application program to implement DCADO functionalities. In further embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, can be implemented as a network, web, distributed, cloud, or other such application accessed on one or more remote computing devices (i.e. servers, cloud, etc.) via Network Interface **25**, such remote computing devices including processing capabilities and instructions for implementing DCADO functionalities. In further embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, can be (1) attached to or interfaced with any computing device or application program, (2) included as a feature of an operating system, (3) built (i.e. hard coded, etc.) into any computing device or application program, and/or (4) available in any other configuration to provide its functionalities.

In some embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, can be implemented at least in part in a computer program such as Java application or program. Java provides a robust and flexible environment for application programs including flexible user interfaces, robust security, built-in network protocols, powerful application programming interfaces, database or DBMS connectivity and interfacing functionalities, file manipulation capabilities, support for networked applications, and/or other features or functionalities. Application programs based on Java can be portable across many devices, yet leverage each device's native capabilities. Java supports the feature sets of most smartphones and a broad range of connected devices while still fitting within their resource constraints. Various Java platforms include virtual machine features comprising a runtime environment for application programs. Java platforms provide a wide range of user-level functionalities that can be implemented in application programs such as displaying text and graphics, playing and recording audio content, displaying and recording visual content, communicating with another computing device, and/or other functionalities. It should be understood that the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, are programming language, platform, and operating system independent. Examples of programming languages that can be used instead of or in addition to Java include C, C++, Cobol, Python, Java Script, Tcl, Visual Basic, Pascal, VB Script, Perl, PHP, Ruby, and/or other programming languages capable of implementing the functionalities described herein.

Where a reference to a specific file or file type is used herein, other files or file types can be substituted.

Where a reference to a data structure is used herein, it should be understood that any variety of data structures can be used such as, for example, array, list, linked list, doubly linked list, queue, tree, heap, graph, map, grid, matrix, multi-dimensional matrix, table, database, database management system (DBMS), file, neural network, and/or any other type or form of a data structure including a custom one. A data structure may include one or more fields or data fields that are part of or associated with the data structure. A field or data field may include a data, an object, a data structure, and/or any other element or a reference/pointer thereto. A data structure can be stored in one or more memories, files, or other repositories. A data structure and/or elements thereof, when stored in a memory, file, or other repository, may be stored in a different arrangement than the arrangement of the data structure and/or elements thereof. For example, a sequence of elements can be stored in an arrangement other than a sequence in a memory, file, or other repository.

Where a reference to a repository is used herein, it should be understood that a repository may be or include one or more files or file systems, one or more storage locations or structures, one or more storage systems, one or more memory locations or structures, and/or other file, storage, memory, or data arrangements.

Where a reference to an interface is used herein, it should be understood that the interface comprises any hardware, device, system, program, method, and/or combination thereof that enable direct or operative coupling, connection, and/or interaction of the elements between which the interface is indicated. A line or arrow shown in the figures between any of the depicted elements comprises such interface. Examples of an interface include a direct connection, an operative connection, a wired connection (i.e. wire, cable, etc.), a wireless connection, a device, a network, a bus, a circuit, a firmware, a driver, a bridge, a program, a combination thereof, and/or others.

Where a reference to an element coupled or connected to another element is used herein, it should be understood that the element may be in communication or other interactive relationship with the other element. Furthermore, an element coupled or connected to another element can be coupled or connected to any other element in alternate implementations. Terms coupled, connected, interfaced, or other such terms may be used interchangeably herein depending on context.

Where a reference to an element matching another element is used herein, it should be understood that the element may be equivalent or similar to the other element. Therefore, the term match or matching can refer to total equivalence or similarity depending on context.

Where a reference to a device is used herein, it should be understood that the device may include or be referred to as a system, and vice versa depending on context, since a device may include a system of elements and a system may be embodied in a device.

Where a reference to a collection of elements is used herein, it should be understood that the collection of elements may include one or more elements. In some aspects or contexts, a reference to a collection of elements does not imply that the collection is an element itself.

Where a reference to an object is used herein, it should be understood that the object may be a physical object (i.e. object detected in a device's surrounding, etc.), an electronic object (i.e. object in an object oriented application program, etc.), and/or other object depending on context.

Where a mention of a function, method, routine, subroutine, or other such procedure is used herein, it should be understood that the function, method, routine, subroutine, or other such procedure comprises a call, reference, or pointer to the function, method, routine, subroutine, or other such procedure.

Where a mention of data, object, data structure, item, element, or thing is used herein, it should be understood that the data, object, data structure, item, element, or thing comprises a reference or pointer to the data, object, data structure, item, element, or thing.

Referring to FIG. **2**, an embodiment of Device **98** comprising Unit for Learning and/or Using a Device's Circumstances for Autonomous Device Operation (DCADO Unit **100**) is illustrated. Device **98** also comprises interconnected Processor **11**, Human-machine Interface **23**, Sensor **92**, Object Processing Unit **93**, Memory **12**, and Storage **27**. Processor **11** includes or executes Application Program **18**. DCADO Unit **100** comprises interconnected Artificial Intelligence Unit **110**, Acquisition Interface **120**, and Modification Interface **130**. Other additional elements can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate embodiments.

In one example, the teaching presented by the disclosure can be implemented in a device or system for learning and/or using a device's circumstances for autonomous device operation. The device or system may include a processor circuit (i.e. Processor **11**, etc.) configured to execute instruction sets (i.e. Instruction Sets **526**, etc.) for operating a device. The device or system may further include a memory unit (i.e. Memory **12**, etc.) configured to store data. The device or system may further include a sensor (i.e. Sensor **92**, etc.) configured to detect objects (i.e. Objects **615**, etc.). The device or system may further include an artificial intelligence unit (i.e. Artificial Intelligence Unit **110**, etc.). The artificial intelligence unit may be configured to receive a first collection of object representations (i.e. Collection of Object Representations **525**, etc.), the first collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may also be configured to receive a first one or more instruction sets for operating the device. The artificial intelligence unit may also be configured to learn the first collection of object representations correlated with the first one or more instruction sets for operating the device. The artificial intelligence unit may also be configured to receive a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may also be configured to anticipate the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may also be configured to cause the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations performed in response to the executing by the processor circuit. Any of the operations of the aforementioned elements can be performed repeatedly and/or in different orders in alternate embodiments. In some embodiments, a stream of collections of object representations can be used instead of or in addition to any collection of object representations such as, for example, using a first stream of collections of object representations instead of the first collection of object representations. In other embodiments, a logic circuit (i.e. Logic Circuit **250**, etc.) may be used instead of the processor circuit. In such embodiments, one or more instruction sets for operating the device (i.e. first one or more instruction sets for operating the device, etc.) may include or be substituted with one or more inputs into or one or more outputs from the logic circuit. In further embodiments, an actuator may be included instead of or in addition to the processor circuit. In such embodiments, one or more instruction sets for operating the device (i.e. first one or more instruction sets for operating the device, etc.) may include or be substituted with one or more inputs into the actuator. Other additional elements can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate embodiments. The device or system for learning and/or using a device's circumstances for autonomous device operation may include any actions or operations of any of the disclosed methods such as methods **9100**, **9200**, **9300**, **9400**, **9500**, **9600**, and/or others (all later described).

Device **98** comprises any hardware, programs, or a combination thereof. Although, Device **98** is referred to as a device herein, Device **98** may be or include a system as a system may be embodied in Device **98**. Device **98** may include any features, functionalities, and embodiments of Computing Device **70**, or elements thereof. In some embodiments, Device **98** includes a computing enabled device for performing mechanical or physical operations (i.e. via actuators, etc.). In other embodiments, Device **98** includes a computing enabled device for performing non-mechanical and/or other operations. Examples of Device **98** include an industrial machine, a toy, a robot, a vehicle, an appliance, a control device, a smartphone or other mobile computer, any computer, and/or other computing enabled device or machine. Such device or machine may be built for any function or purpose some examples of which are described later.

User **50** (also referred to simply as user or other suitable name or reference) comprises a human user or non-human user. A non-human User **50** includes any device, system, program, and/or other mechanism for operating or controlling Device **98** and/or elements thereof. In one example, User **50** may issue an operating direction to Application Program **18** responsive to which Application Program's **18** instructions or instruction sets may be executed by Processor **11** to perform a desired operation on Device **98**. In another example, User **50** may issue an operating direction to Processor **11**, Logic Circuit **250** (later described), and/or other processing element responsive to which Processor **11**, Logic Circuit **250**, and/or other processing element may implement logic to perform a desired operation on Device **98**. User's **50** operating directions comprise any user inputted data (i.e. values, text, symbols, etc.), directions (i.e. move right, move up, move forward, copy an item, click on a link, etc.), instructions or instruction sets (i.e. manually inputted instructions or instruction sets, etc.), and/or other inputs or information. A non-human User **50** can utilize more suitable interfaces instead of, or in addition to, Human-machine Interface **23** and/or Display **21** for controlling Device **98** and/or elements thereof. Examples of such interfaces include an application programming interface (API), bridge (i.e. bridge between applications, devices, or systems, etc.), driver, socket, direct or operative connection, handle, function/routine/subroutine, and/or other interfaces.

In some embodiments, Processor **11**, Logic Circuit **250**, Application Program **18**, and/or other processing element may control or affect an actuator (not shown). Actuator comprises the functionality for implementing motion,

actions, behaviors, maneuvers, and/or other mechanical or physical operations. Device **98** may include one or more actuators to enable Device **98** to perform mechanical, physical, or other operations and/or to interact with its environment. For example, an actuator may include or be coupled to an element such as a wheel, arm, or other element to act upon the environment. Examples of an actuator include a motor, a linear motor, a servomotor, a hydraulic element, a pneumatic element, an electro-magnetic element, a spring element, and/or other actuators. Examples of types of actuators include a rotary actuator, a linear actuator, and/or other types of actuators. In other embodiments, Processor **11**, Logic Circuit **250**, Application Program **18**, and/or other processing element may control or affect any other device or element instead of or in addition to an actuator.

Referring to FIGS. **3**A-**3**E, various embodiments of Sensors **92** and elements of Object Processing Unit **93** are illustrated.

Sensor **92** (also referred to as simply as sensor or other suitable name or reference) comprises the functionality for obtaining or detecting information about its environment, and/or other functionalities. As such, one or more Sensors **92** can be used to detect objects and/or their properties in Device's **98** surrounding. In some aspects, Device's **98** surrounding may include exterior of Device **98**. In other aspects, Device's **98** surrounding may include interior of Device **98** in case of hollow Device **98**, Device **98** comprising compartments or openings, and/or other variously shaped Device **98**. Examples of aspects of an environment that Sensor **92** can measure or be sensitive to include light (i.e. camera, lidar, etc.), electromagnetism/electromagnetic field (i.e. radar, etc.), sound (i.e. microphone, sonar, etc.), physical contact (i.e. tactile sensor, etc.), magnetism/magnetic field (i.e. compass, etc.), electricity/electric field, temperature, gravity, vibration, pressure, and/or others. In some aspects, a passive sensor (i.e. camera, microphone, etc.) measures signals or radiation emitted or reflected by an object. In other aspects, an active sensor (i.e. lidar, radar, sonar, etc.) emits signals or radiation and measures the signals or radiation reflected or backscattered from an object. A reference to a Sensor **92** herein includes a reference to one or more Sensors **92** as applicable. In some designs, a plurality of Sensors **92** may be used to detect objects and/or their properties from different angles or sides of Device **98**. For example, four Cameras **92***a* can be placed on four corners of Device **98** to cover 360 degrees of view of Device's **98** surrounding. In other designs, a plurality of different types of Sensors **92** may be used to detect different types of objects and/or their properties. For example, one or more Cameras **92***a* can be used to detect and identify an object, whereas, Radar **92***d* can be used to determine distance and bearing/angle of the object relative to Device **98**. In further designs, a signal-emitting element can be placed within or onto an object and Sensor **92** can detect the signal from the signal-emitting element, thereby detecting the object and/or its properties. For example, a radio-frequency identification (RFID) emitter may be placed within an object to help Sensor **92** detect, identify, and/or obtain other information about the object.

In some embodiments, Sensor **92** may be or include Camera **92***a* as shown in FIG. **3**A. Camera **92***a* comprises the functionality for capturing one or more pictures, and/or other functionalities. As such, Camera **92***a* can be used to capture pictures of Device's **98** surrounding. Camera **92***a* may be useful in detecting existence of an object, type of an object, identity of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In some aspects, Camera **92***a* may be or comprises a motion picture camera that can capture streams of pictures (i.e. motion pictures, videos, etc.). In other aspects, Camera **92***a* may be or comprises a still picture camera that can capture still pictures (i.e. photographs, etc.). In further aspects, Camera **92***a* may be or comprises a stereo camera (i.e. camera with multiple lenses, etc.) that can capture stereoscopic or range pictures. In further aspects, Camera **92***a* may be or comprises any other Camera **92***a*. In general, Camera **92***a* may capture any light (i.e. visible light, infrared light, ultraviolet light, x-ray light, etc.) across the electromagnetic spectrum onto a light-sensitive material. Any other technique known in art can be utilized to facilitate Camera **92***a* functionalities. In one example, a digital Camera **92***a* can utilize a charge coupled device (CCD), a complementary metal—oxide—semiconductor (CMOS) sensor, and/or other electronic image sensor to capture digital pictures that can then be stored in a memory or storage, or transmitted to any of the disclosed or other elements for further processing. In another example, analog Camera **92***a* can utilize an analog-to-digital converter to produce digital pictures. In some embodiments, Camera **92***a* can be built, embedded, or integrated in Device **98** and/or other disclosed element. In other embodiments, Camera **92***a* can be an external Camera **92***a* connected with Device **98** and/or other disclosed element. In further embodiments, Camera **92***a* comprises Computing Device **70** or elements thereof. In general, Camera **92***a* can be implemented in any suitable configuration to provide its functionalities. Camera **92***a* may capture one or more digital pictures. A digital picture may include a collection of color encoded pixels or dots. Examples of file formats that can be utilized to store a digital picture include JPEG, GIF, TIFF, PNG, PDF, and/or other digitally encoded picture formats. A stream of digital pictures (i.e. motion picture, video, etc.) may include one or more digital pictures. Examples of file formats that can be utilized to store a stream of digital pictures include MPEG, AVI, FLV, MOV, RM, SWF, WMV, DivX, and/or other digitally encoded motion picture formats.

In other embodiments, Sensor **92** may be or include Microphone **92***b* as shown in FIG. **3**B. Microphone **92***b* comprises the functionality for capturing one or more sounds, and/or other functionalities. As such, Microphone **92***b* can be used to capture sounds from Device's **98** surrounding. Microphone **92***b* may be useful in detecting existence of an object, type of an object, identity of an object, bearing/angle of an object, activity (i.e. motion, sounding, etc.) of an object, and/or other properties of an object. In some aspects, Microphone **92***b* may be omnidirectional microphone that enables capturing sounds from any direction. In other aspects, Microphone **92***b* may be a directional (i.e. unidirectional, bidirectional, etc.) microphone that enables capturing sounds from one or more directions while ignoring or being insensitive to sounds from other directions. In general, Microphone **92***b* may utilize a membrane sensitive to air pressure and may produce electrical signal from air pressure variations. Samples of the electrical signal can then be read to produce a stream of digital sound samples. Any other technique known in art can be utilized to facilitate Microphone **92***b* functionalities. In one example, a digital Microphone **92***b* may include an integrated analog-to-digital converter to capture a stream of digital sound samples that can then be stored in a memory or storage, or transmitted to any of the disclosed or other elements for further processing. In another example, analog Microphone **92***b* may utilize an external analog-to-digital

converter to produce a stream of digital sound samples. In some embodiments, Microphone **92***b* can be built, embedded, or integrated in Device **98**. In other embodiments, Microphone **92***b* can be an external Microphone **92***b* connected with Device **98**. In further embodiments where used in water, Microphone **92***b* may be or include a hydrophone. In further embodiments, Microphone **92***b* comprises Computing Device **70** or elements thereof. In general, Microphone **92***b* can be implemented in any suitable configuration to provide its functionalities. Examples of file formats that can be utilized to store a stream of digital sound samples include WAV, WMA, AIFF, MP3, RA, OGG, and/or other digitally encoded sound formats.

In further embodiments, Sensor **92** may be or include Lidar **92***c* as shown in FIG. **3**C. Lidar **92***c* may be useful in detecting existence of an object, type of an object, identity of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In some aspects, Lidar **92***c* may emit a light signal (i.e. laser beam, etc.) and listen for a signal that is reflected or backscattered from an object. Any other technique known in art can be utilized to facilitate Lidar **92***c* functionalities.

In further embodiments, Sensor **92** may be or include a Radar **92***d* as shown in FIG. **3**D. Radar **92***d* may be useful in detecting existence of an object, type of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In some aspects, Radar **92***d* may emit a radio signal (i.e. radio wave, etc.) and listen for a signal that is reflected or backscattered from an object. Any other technique known in art can be utilized to facilitate Radar **92***d* functionalities.

In further embodiments, Sensor **92** may be or include Sonar **92***e* as shown in FIG. **3**E. Sonar **92***e* may be useful in detecting existence of an object, type of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In some aspects, Sonar **92***e* may emit a sound signal (i.e. sound pulse, etc.) and listen for a signal that is reflected or backscattered from an object. Any other technique known in art can be utilized to facilitate Sonar **92***e* functionalities.

One of ordinary skill in art will understand that the aforementioned sensors are described merely as examples of a variety of possible implementations, and that while all possible sensors are too voluminous to describe, other sensors known in art that can facilitate detecting of objects and/or their properties in Device's **98** surrounding are within the scope of this disclosure. Any combination of the aforementioned and/or other sensors can be used in various embodiments.

Object Processing Unit **93** comprises the functionality for processing output from Sensor **92** to obtain information of interest, and/or other functionalities. As such, Object Processing Unit **93** can be used to process output from Sensor **92** to detect objects and/or their properties in Device's **98** surrounding. In some embodiments, Object Processing Unit **93** comprises the functionality for creating or generating Collection of Object Representations **525** (also referred to as Coll of Obj Rep or other suitable name or reference) and storing one or more Object Representations **625** (also referred to simply as object representations, representations of objects, or other suitable name or reference), Object Properties **630** (also referred to simply as object properties or other suitable name or reference), and/or other elements or information into the Collection of Object Representations **525**. As such, Collection of Object Representations **525** comprises the functionality for storing one or more Object Representations **625**, Object Properties **630**, and/or other elements or information. Object Representation **625** may include an electronic representation of an object (i.e. Object **615** [later described], etc.) detected in Device's **98** surrounding. In some aspects, Collection of Object Representations **525** includes one or more Object Representations **625**, Object Properties **630**, and/or other elements or information related to objects detected in Device's **98** surrounding at a particular time. Collection of Object Representations **525** may, therefore, include knowledge (i.e. unit of knowledge, etc.) of Device's **98** circumstances including objects with various properties at a particular time. In some designs, a Collection of Object Representations **525** may include or be associated with a time stamp (not shown), order (not shown), or other time related information. For example, one Collection of Object Representations **525** may be associated with time stamp t1, another Collection of Object Representations **525** may be associated with time stamp t2, and so on. Time stamps t1, t2, etc. may indicate the times of generating Collections of Object Representations **525**, for instance. In other embodiments, Object Processing Unit **93** comprises the functionality for creating or generating a stream of Collections of Object Representations **525**. A stream of Collections of Object Representations **525** may include one Collection of Object Representations **525** or a group, sequence, or other plurality of Collections of Object Representations **525**. In some aspects, a stream of Collections of Object Representations **525** includes one or more Collections of Object Representations **525**, and/or other elements or information related to objects detected in Device's **98** surrounding over time. A stream of Collections of Object Representations **525** may, therefore, include knowledge (i.e. unit of knowledge, etc.) of Device's **98** circumstances including objects with various properties over time. As circumstances including objects with various properties in Device's **98** surrounding change (i.e. objects and/or their properties change, move, act, transform, etc.) over time, this change may be captured in a stream of Collections of Object Representations **525**. In some designs, each Collection of Object Representations **525** in a stream may include or be associated with the aforementioned time stamp, order, or other time related information. For example, one Collection of Object Representations **525** in a stream may be associated with order **1**, a next Collection of Object Representations **525** in the stream may be associated with order **2**, and so on. Orders **1**, **2**, etc. may indicate the orders or places of Collections of Object Representations **525** within a stream (i.e. sequence, etc.), for instance. Examples of objects include biological objects (i.e. persons, animals, vegetation, etc.), nature objects (i.e. rocks, bodies of water, etc.), man-made objects (i.e. buildings, streets, ground/aerial/aquatic vehicles, etc.), and/or others. In some aspects, any part of an object may be detected as an object itself. For instance, instead of or in addition to detecting a vehicle as an object, a wheel and/or other parts of the vehicle may be detected as objects. In general, object may include any object or part thereof that can be detected. Examples of object properties include existence of an object, type of an object (i.e. person, cat, vehicle, building, street, tree, rock, etc.), identity of an object (i.e. name, identifier, etc.), distance of an object, bearing/angle of an object, location of an object (i.e. distance and bearing/angle from a known point, coordinates, etc.), shape/size of an object (i.e. height, width, depth, computer model, point cloud, etc.), activity of an object (i.e. motion, gestures, etc.), and/or other properties of an object. Type of

an object, for example, may include any classification of objects ranging from detailed such as person, cat, vehicle, building, street, tree, rock, etc. to generalized such as biological object, nature object, manmade object, etc., and/or others including their sub-types. Location of an object, for example, can include a relative location such as one defined by distance and bearing/angle from a known point or location (i.e. Device **98** location, etc.). Location of an object, for example, can also include absolute location such as one defined by object coordinates. In general, an object property may include any attribute of an object (i.e. existence of an object, type of an object, identity of an object, shape/size of an object, etc.), any relationship of an object with Device **98**, other objects, or the environment (i.e. distance of an object, bearing/angle of an object, friend/foe relationship, etc.), and/or other information related to an object. In some implementations, Object Processing Unit **93** and/or any of its elements or functionalities can be included in Sensor **92**. In other implementations, Object Processing Unit **93** and/or any of its elements or functionalities can be embedded into or operate on Processor **11**. In further implementations, Object Processing Unit **93** and/or any of its elements or functionalities can be embedded into or operate in DCADO Unit **100**, and/or other disclosed elements. Object Processing Unit **93** may be provided in any suitable configuration. Object Processing Unit **93** may include any signal processing techniques or elements known in art as applicable.

In some embodiments, Object Processing Unit **93** may include Picture Recognizer **94*a*** as shown in FIG. **3**A. Picture Recognizer **94*a*** comprises the functionality for detecting or recognizing objects and/or their properties in visual data, and/or other disclosed functionalities. Visual data includes digital motion pictures, digital still pictures, and/or other visual data. Examples of file formats that can be utilized to store visual data include AVI, DivX, MPEG, JPEG, GIF, TIFF, PNG, PDF, and/or other file formats. For example, Picture Recognizer **94*a*** can be used for detecting or recognizing objects and/or their properties in one or more digital pictures captured by one or more Cameras **92*a***. Picture Recognizer **94*a*** can be utilized in detecting or recognizing existence of an object, type of an object, identity of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In general, Picture Recognizer **94*a*** can be used for any operation supported by Picture Recognizer **94*a***. Picture Recognizer **94*a*** may detect or recognize an object and/or its properties as well as track the object and/or its properties in one or more digital pictures or streams of digital pictures (i.e. motion pictures, video, etc.). In the case of a person, Picture Recognizer **94*a*** may detect or recognize a human head or face, upper body, full body, or portions/combinations thereof. In some aspects, Picture Recognizer **94*a*** may detect or recognize objects and/or their properties from a digital picture by comparing regions of pixels from the digital picture with collections of pixels comprising known objects and/or their properties. The collections of pixels comprising known objects and/or their properties can be learned or manually, programmatically, or otherwise defined. The collections of pixels comprising known objects and/or their properties can be stored in any data structure or repository (i.e. one or more files, database, etc.) that resides locally on Device **98**, or remotely on a remote computing device (i.e. server, cloud, etc.) accessible over a network or an interface. In other aspects, Picture Recognizer **94*a*** may detect or recognize objects and/or their properties from a digital picture by comparing features (i.e. lines, edges, ridges, corners, blobs, regions, etc.) of the digital picture with features of known objects and/or their properties. The features of known objects and/or their properties can be learned or manually, programmatically, or otherwise defined. The features of known objects and/or their properties can be stored in any data structure or repository (i.e. neural network, one or more files, database, etc.) that resides locally on Device **98**, or remotely on a remote computing device (i.e. server, cloud, etc.) accessible over a network or an interface. Typical steps or elements in a feature oriented picture recognition include pre-processing, feature extraction, detection/segmentation, decision-making, and/or others, or a combination thereof, each of which may include its own sub-steps or sub-elements depending on the application. In further aspects, Picture Recognizer **94*a*** may detect or recognize multiple objects and/or their properties from a digital picture using the aforementioned pixel or feature comparisons, and/or other detection or recognition techniques. For example, a picture may depict two objects in two of its regions both of which Picture Recognizer **94*a*** can detect simultaneously. In further aspects, where objects and/or their properties span multiple pictures, Picture Recognizer **94*a*** may detect or recognize objects and/or their properties by applying the aforementioned pixel or feature comparisons and/or other detection or recognition techniques over a stream of digital pictures (i.e. motion picture, video, etc.). For example, once an object is detected in a digital picture (i.e. frame, etc.) of a stream of digital pictures (i.e. motion picture, video, etc.), the region of pixels comprising the detected object or the object's features can be searched in other pictures of the stream of digital pictures, thereby tracking the object through the stream of digital pictures. In further aspects, Picture Recognizer **94*a*** may detect or recognize an object's activities by identifying and/or analyzing differences between a detected region of pixels of one picture (i.e. frame, etc.) and detected regions of pixels of other pictures in a stream of digital pictures. For example, a region of pixels comprising a person's face can be detected in multiple consecutive pictures of a stream of digital pictures (i.e. motion picture, video, etc.). Differences among the detected regions of the consecutive pictures may be identified in the mouth part of the person's face to indicate smiling or speaking activity. In further aspects, Picture Recognizer **94*a*** may detect or recognize objects and/or their properties using one or more artificial neural networks, which may include statistical techniques. Examples of artificial neural networks that can be used in Picture Recognizer **94*a*** include convolutional neural networks (CNNs), time delay neural networks (TDNNs), deep neural networks, and/or others. In one example, picture recognition techniques and/or tools involving convolutional neural networks may include identifying and/or analyzing tiled and/or overlapping regions or features of a digital picture, which may then be used to search for pictures with matching regions or features. In another example, features of different convolutional neural networks responsible for spatial and temporal streams can be fused to detect objects and/or their properties in streams of digital pictures (i.e. motion pictures, videos, etc.). In general, Picture Recognizer **94*a*** may include any machine learning, deep learning, and/or other artificial intelligence techniques. In further aspects, Picture Recognizer **94*a*** can detect distance of a recognized object in a picture captured by a camera using structured light, sheet of light, or other lighting schemes, and/or by using phase shift analysis, time of flight, interferometry, or other techniques. In further aspects, Picture Recognizer **94*a*** may detect distance of a recognized object in a picture captured by a stereo camera by

using triangulation and/or other techniques. In further aspects, Picture Recognizer **94***a* may detect bearing/angle of a recognized object relative to the camera-facing direction by measuring the distance from the vertical centerline of the picture to a pixel in the recognized object based on known picture resolution and camera's angle of view. Any other techniques known in art can be utilized in Picture Recognizer **94***a*. For example, thresholds for similarity, statistical techniques, and/or optimization techniques can be utilized to determine a match in any of the above-described detection or recognition techniques. In some exemplary embodiments, object recognition techniques and/or tools such as OpenCV (Open Source Computer Vision) library, CamFind API, Kooaba, 6px API, Dextro API, and/or others can be utilized for detecting or recognizing objects and/or their properties in digital pictures. In some aspects, picture recognition techniques and/or tools involve identifying and/or analyzing features such as lines, edges, ridges, corners, blobs, regions, and/or their relative positions, sizes, shapes, etc., which may then be used to search for pictures with matching features. For example, OpenCV library can detect an object (i.e. person, animal, vehicle, rock, etc.) and/or its properties in one or more digital pictures captured by Camera **92***a* or stored in an electronic repository, which can then be utilized in DCADO Unit **100**, Artificial Intelligence Unit **110**, and/or other elements. In other exemplary embodiments, facial recognition techniques and/or tools such as OpenCV (Open Source Computer Vision) library, Animetrics FaceR API, Lambda Labs Facial Recognition API, Face++SDK, Neven Vision (also known as N-Vision) Engine, and/or others can be utilized for detecting or recognizing faces in digital pictures. In some aspects, facial recognition techniques and/or tools involve identifying and/or analyzing facial features such as the relative position, size, and/or shape of the eyes, nose, cheekbones, jaw, etc., which may then be used to search for pictures with matching features. For example, FaceR API can detect a person's face in one or more digital pictures captured by Camera **92***a* or stored in an electronic repository, which can then be utilized in DCADO Unit **100**, Artificial Intelligence Unit **110**, and/or other elements.

Various aspects or properties of digital pictures or pixels can be taken into account by Picture Recognizer **94***a* in any of the recognizing or comparisons. Examples of such aspects or properties include color adjustment, size adjustment, content manipulation, transparency (i.e. alpha channel, etc.), use of mask, and/or others. In some implementations, as digital pictures can be captured by various picture taking equipment, in various environments, and under various lighting conditions, Picture Recognizer **94***a* can adjust lighting or color of pixels or otherwise manipulate pixels before or during comparison. Lighting or color adjustment (also referred to as gray balance, neutral balance, white balance, etc.) may generally include manipulating or rebalancing the intensities of the colors (i.e. red, green, and/or blue if RGB color model is used, etc.) of one or more pixels. For example, Picture Recognizer **94***a* can adjust lighting or color of some or all pixels of one picture to make it more comparable to another picture. Picture Recognizer **94***a* can also incrementally adjust the pixels such as increasing or decreasing the red, green, and/or blue pixel values by a certain amount in each cycle of comparisons in order to find a substantially similar match at one of the incremental adjustment levels. Any of the publically available, custom, or other lighting or color adjustment techniques or programs can be utilized such as color filters, color balancing, color correction, and/or others. In other implementations, Picture Recognizer **94***a* can resize or otherwise transform a digital picture before or during comparison. Such resizing or transformation may include increasing or decreasing the number of pixels of a digital picture. For example, Picture Recognizer **94***a* can increase or decrease the size of a digital picture proportionally (i.e. increase or decrease length and/or width keeping aspect ratio constant, etc.) to equate its size with the size of another digital picture. Picture Recognizer **94***a* can also incrementally resize a digital picture such as increasing or decreasing the size of the digital picture proportionally by a certain amount in each cycle of comparisons in order to find a substantially similar match at one of the incremental sizes. Any of the publically available, custom, or other digital picture resizing techniques or programs can be utilized such as nearest-neighbor interpolation, bilinear interpolation, bicubic interpolation, and/or others. In further implementations, Picture Recognizer **94***a* can manipulate content (i.e. all pixels, one or more regions, one or more depicted objects, etc.) of a digital picture before or during comparison. Such content manipulation may include moving, centering, aligning, resizing, transforming, and/or otherwise manipulating content of a digital picture. For example, Picture Recognizer **94***a* can move, center, or align content of one picture to make it more comparable to another picture. Any of the publically available, custom, or other digital picture manipulation techniques or programs can be utilized such as pixel moving, warping, distorting, aforementioned interpolations, and/or others. In further implementations, in digital pictures comprising transparency features or functionalities, Picture Recognizer **94***a* can utilize a threshold for acceptable number or percentage transparency difference. Alternatively, transparency can be applied to one or more pixels of a digital picture and color difference may then be determined between compared pixels taking into account the transparency related color effect. Alternatively, transparent pixels can be excluded from comparison. In further implementations, certain regions or subsets of pixels can be ignored or excluded during comparison using a mask. In general, any region or subset of a picture determined to contain no content of interest can be excluded from comparison using a mask. Examples of such regions or subsets include background, transparent or partially transparent regions, regions comprising insignificant content, or any arbitrary region or subset. Picture Recognizer **94***a* can perform any other pre-processing or manipulation of digital pictures or pixels before or during recognizing or comparison.

In other embodiments, Object Processing Unit **93** may include Sound Recognizer **94***b* as shown in FIG. **3**B. Sound Recognizer **94***b* comprises the functionality for detecting or recognizing objects and/or their properties in audio data, and/or other disclosed functionalities. Audio data includes digital sound, and/or other audio data. Examples of file formats that can be utilized to store audio data include WAV, WMA, AIFF, MP3, RA, OGG, and/or other file formats. For example, Sound Recognizer **94***b* can be used for detecting or recognizing objects and/or their properties in a stream of digital sound samples captured by one or more Microphones **92***b*. In the case of a person, Sound Recognizer **94***b* may detect or recognize human voice. Sound Recognizer **94***b* can be utilized in detecting or recognizing existence of an object, type of an object, identity of an object, bearing/angle of an object, activity (i.e. motion, sounding, etc.) of an object, and/or other properties of an object. In general, Sound Recognizer **94***b* can be used for any operation supported by Sound Recognizer **94***b*. In some aspects, Sound Recognizer **94***b* may detect or recognize an object and/or its properties

from a stream of digital sound samples by comparing collections of sound samples from the stream of digital sound samples with collections of sound samples of known objects and/or their properties. The collections of sound samples of known objects and/or their properties can be learned, or manually, programmatically, or otherwise defined. The collections of sound samples of known objects and/or their properties can be stored in any data structure or repository (i.e. one or more files, database, etc.) that resides locally on Device **98**, or remotely on a remote computing device (i.e. server, cloud, etc.) accessible over a network or an interface. In other aspects, Sound Recognizer **94***b* may detect or recognize an object and/or its properties from a stream of digital sound samples by comparing features from the stream of digital sound samples with features of sounds of known objects and/or their properties. The features of sounds of known objects and/or their properties can be learned, or manually, programmatically, or otherwise defined. The features of sounds of known objects and/or their properties can be stored in any data structure or repository (i.e. one or more files, database, neural network, etc.) that resides locally on Device **98**, or remotely on a remote computing device (i.e. server, cloud, etc.) accessible over a network or an interface. Typical steps or elements in a feature oriented sound recognition include pre-processing, feature extraction, acoustic modeling, language modeling, and/or others, or a combination thereof, each of which may include its own sub-steps or sub-elements depending on the application. In further aspects, Sound Recognizer **94***b* may detect or recognize a variety of sounds from a stream of digital sound samples using the aforementioned sound sample or feature comparisons, and/or other detection or recognition techniques. For example, sound of a person, animal, vehicle, and/or other sounds can be detected by Sound Recognizer **94***b*. In further aspects, Sound Recognizer **94***b* may detect or recognize sounds using Hidden Markov Models (HMM), Artificial Neural Networks, Dynamic Time Warping (DTW), Gaussian Mixture Models (GMM), and/or other models or techniques, or a combination thereof. Some or all of these models or techniques may include statistical techniques. Examples of artificial neural networks that can be used in Sound Recognizer **94***b* include recurrent neural networks, time delay neural networks (TDNNs), deep neural networks, convolutional neural networks, and/or others. In general, Sound Recognizer **94***b* may include any machine learning, deep learning, and/or other artificial intelligence techniques. In further aspects, Sound Recognizer **94***b* may detect bearing/angle of a recognized object by measuring the direction in which Microphone **92***b* is pointing when sound of maximum strength is received, by analyzing amplitude of the sound, by performing phase analysis (i.e. with microphone array, etc.) of the sound, and/or by utilizing other techniques. Any other techniques known in art can be utilized in Sound Recognizer **94***b*. For example, thresholds for similarity, statistical techniques, and/or optimization techniques can be utilized to determine a match in any of the above-described detection or recognition techniques. In some exemplary embodiments, operating system's Sound recognition functionalities such as iOS's Voice Services, Siri, and/or others can be utilized in Sound Recognizer **94***b*. For example, iOS Voice Services can detect an object (i.e. person, etc.) and/or its properties in a stream of digital sound samples captured by Microphone **92***b* or stored in an electronic repository, which can then be utilized in DCADO Unit **100**, Artificial Intelligence Unit **110**, and/or other elements. In other exemplary embodiments, Java Speech API (JSAPI) implementation such as The Cloud Garden, Sphinx, and/or others can be utilized in Sound Recognizer **94***b*. For example, Cloud Garden JSAPI can detect an object (i.e. person, animal, vehicle, etc.) and/or its properties in a stream of digital sound samples captured by Microphone **92***b* or stored in an electronic repository, which can then be utilized in DCADO Unit **100**, Artificial Intelligence Unit **110**, and/or other elements. Any other programming language's or platform's speech or sound processing API can similarly be utilized. In further exemplary embodiments, applications or engines providing Sound recognition functionalities such as HTK (Hidden Markov Model Toolkit), Kaldi, OpenEars, Dragon Mobile, Julius, iSpeech, CeedVocal, and/or others can be utilized in Sound Recognizer **94***b*. For example, Kaldi SDK can detect an object (i.e. person, animal, vehicle, etc.) and/or its properties in a stream of digital sound samples captured by Microphone **92***b* or stored in an electronic repository, which can then be utilized in DCADO Unit **100**, Artificial Intelligence Unit **110**, and/or other elements.

In further embodiments, Object Processing Unit **93** may include Lidar Processing Unit **94***c* as shown in FIG. **3**C. Lidar Processing Unit **94***c* comprises the functionality for detecting or recognizing objects and/or their properties using light, and/or other disclosed functionalities. As such, Lidar Processing Unit **94***c* can be utilized in detecting existence of an object, type of an object, identity of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In general, Lidar Processing Unit **94***c* can be used for any operation supported by Lidar Processing Unit **94***c*. In one example, Lidar Processing Unit **94***c* may detect distance of an object by measuring time delay between emission of a light signal (i.e. laser beam, etc.) and return of the light signal reflected from the object based on known speed of light. In another example, Lidar Processing Unit **94***c* may detect bearing/angle of an object by analyzing the amplitudes of a light signal received by an array of detectors (i.e. detectors arranged into a quadrant or other arrangement, etc.). In a further example, Lidar Processing Unit **94***c* may detect existence, type, identity, shape/size, activity, and/or other properties of an object by illuminating the object with light and acquiring an image of the object, which can then be processed using some of the previously described or other picture recognition techniques. In a further example, Lidar Processing Unit **94***c* may detect existence, type, identity, shape/size, activity, and/or other properties of an object by illuminating the object with light and acquiring a point cloud representation of the object. Lidar Processing Unit **94***c* may detect objects and/or their properties by utilizing any lidar or light-related techniques known in art.

In further embodiments, Object Processing Unit **93** may include Radar Processing Unit **94***d* as shown in FIG. **3**D. Radar Processing Unit **94***d* comprises the functionality for detecting or recognizing objects and/or their properties using radio waves, and/or other disclosed functionalities. As such, Radar Processing Unit **94***d* can be utilized in detecting existence of an object, type of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In general, Radar Processing Unit **94***d* can be used for any operation supported by Radar Processing Unit **94***d*. In one example, Radar Processing Unit **94***d* may detect existence of an object by emitting a radio signal and listening for the radio signal reflected from the object. In another example, Radar Processing Unit **94***d* may detect distance of an object by measuring time delay

between emission of a radio signal and return of the radio signal reflected from the object based on known speed of the radio signal. In a further example, Radar Processing Unit **94***d* may detect bearing/angle of an object by measuring the direction in which the antenna is pointing when the return signal of maximum strength is received, by analyzing amplitude of the return signal, by performing phase analysis (i.e. with antenna array, etc.) of the return signal, and/or by utilizing any amplitude, phase, or other techniques. In a further example, Radar Processing Unit **94***d* may detect existence, type, identity, shape/size, activity, and/or other properties of an object by illuminating the object with radio waves and acquiring an image of the object, which can then be processed using some of the previously described or other picture recognition techniques. Radar Processing Unit **94***d* may detect objects and/or their properties by utilizing any radar or radio-related techniques known in art.

In further embodiments, Object Processing Unit **93** may include Sonar Processing Unit **94***e* as shown in FIG. **3**E. Sonar Processing Unit **94***e* comprises the functionality for detecting or recognizing objects and/or their properties using sound, and/or other disclosed functionalities. As such, Sonar Processing Unit **94***e* can be utilized in detecting existence of an object, type of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In general, Sonar Processing Unit **94***e* can be used for any operation supported by Sonar Processing Unit **94***e*. In one example, Sonar Processing Unit **94***e* may detect existence of an object by emitting a sound signal and listening for the sound signal reflected from the object. In another example, Sonar Processing Unit **94***e* may detect distance of an object by measuring time delay between emission of a sound signal and return of the sound signal reflected from the object based on known speed of the sound signal. In a further example, Sonar Processing Unit **94***e* may detect bearing/angle of an object by measuring the direction in which the microphone is pointing when the return signal of maximum strength is received, by analyzing amplitude of the return signal, by performing phase analysis (i.e. with microphone array, etc.) of the return signal, and/or by utilizing any amplitude, phase, or other techniques. In a further example, Sonar Processing Unit **94***e* may detect existence, type, identity, shape/size, activity, and/or other properties of an object by illuminating the object with sound pulses and acquiring an image of the object, which can then be processed using some of the previously described or other picture recognition techniques. Sonar Processing Unit **94***e* may detect objects and/or their properties by utilizing any sonar or sound-related techniques known in art.

One of ordinary skill in art will understand that the aforementioned techniques for detecting or recognizing objects and/or their properties are described merely as examples of a variety of possible implementations, and that while all possible techniques for detecting or recognizing objects and/or their properties are too voluminous to describe, other techniques for detecting or recognizing objects and/or their properties known in art are within the scope of this disclosure. Also, any signal processing technique known in art that can facilitate the disclosed functionalities can be utilized in various embodiments. Any combination of the aforementioned and/or other sensors, object detecting or recognizing techniques, signal processing techniques, and/or other elements or techniques can be used in various embodiments.

Referring to FIGS. **4**A-**4**B, an exemplary embodiment of Objects **615** (also referred to simply as objects or other suitable name or reference) detected in Device's **98** surrounding, and resulting Collection of Object Representations **525** are illustrated.

As shown for example in FIG. **4**A, Object **615***a* is detected. Object **615***a* may be recognized as a cat. Object **615***a* may be detected at a distance of 6 m from Device **98**. Object **615***a* may be detected at a bearing/angle of 56° from Device's **98** centerline. Furthermore, Object **615***b* is also detected. Object **615***b* may be recognized as a tree. Object **615***b* may be detected at a distance of 10 m from Device **98**. Object **615***b* may be detected at a bearing/angle of 131° from Device's **98** centerline. Furthermore, Object **615***c* is also detected. Object **615***c* may be recognized as a person. Object **615***c* may be identified as John Doe. Object **615***c* may be detected at a distance of 8 m from Device **98**. Object **615***c* may be detected at a bearing/angle of 287° from Device's **98** centerline. Any other Objects **615** instead of or in addition to Object **615***a*, Object **615***b*, and Object **615***c* may be detected. In some aspects, any features, functionalities, and embodiments of Camera **92***a*/Picture Recognizer **94***a*, Microphone **92***b*/Sound Recognizer **94***b*, and/or other sensors or techniques can be utilized for recognizing and/or identifying a person, a cat, a tree, and/or other Objects **615**. In further aspects, any features, functionalities, and embodiments of Camera **92***a*/Picture Recognizer **94***a*, Microphone **92***b*/Sound Recognizer **94***b*, Lidar **92***c*/Lidar Processing Unit **94***c*, Radar **92***d*/Radar Processing Unit **94***d*, Sonar **92***e*/Sonar Processing Unit **94***e*, and/or other sensors or techniques can be utilized for detecting distance, bearing/angle, and/or other object properties.

As shown for example in FIG. **4**B, Object Processing Unit **93** may create or generate Collection of Object Representations **525** including Object Representation **625***a* representing Object **615***a*, Object Representation **625***b* representing Object **615***b*, Object Representation **625***c* representing Object **615***c*, etc. For instance, Object Representation **625***a* may include Object Property **630***aa* "Cat" in Category **635***aa* "Type", Object Property **630***ab* "6 m" in Category **635***ab* "Distance", Object Property **630***ac* "56°" in Category **635***ac* "Bearing", etc. Also, Object Representation **625***b* may include Object Property **630***ba* "Tree" in Category **635***ba* "Type", Object Property **630***bb* "10 m" in Category **635***bb* "Distance", Object Property **630***bc* "131°" in Category **635***bc* "Bearing", etc. Also, Object Representation **625***c* may include Object Property **630***ca* "Person" in Category **635***ca* "Type", Object Property **630***cb* "John Doe" in Category **635***cb* "Identity", Object Property **630***cc* "8 m" in Category **635***cc* "Distance", Object Property **630***cd* "287°" in Category **635***cd* "Bearing", etc. Any number of Object Representations **625**, and/or other elements or information can be included in Collection of Object Representations **525**. Any number of Object Properties **630** (also referred to simply as object properties or other suitable name or reference), and/or other elements or information can be included in an Object Representation **625**. In some aspects, a reference to Collection of Object Representations **525** comprises a reference to a collection of Object Properties **630** and/or other elements or information related to one or more Objects **615**. Other additional Object Representations **625**, Object Properties **630**, elements, and/or information can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate embodiments of Collection of Object Representations **525**.

Referring now to DCADO Unit **100**, DCADO Unit **100** comprises any hardware, programs, or a combination thereof. DCADO Unit **100** comprises the functionality for

learning the operation of Device **98** in circumstances including objects with various properties. DCADO Unit **100** comprises the functionality for structuring and/or storing this knowledge in a knowledgebase (i.e. neural network, graph, sequences, other repository, etc.). DCADO Unit **100** comprises the functionality for enabling autonomous operation of Device **98** in circumstances including objects with various properties. DCADO Unit **100** comprises the functionality for interfacing with or attaching to Application Program **18**, Processor **11**, Logic Circuit **250** (later described), and/or other processing element. DCADO Unit **100** comprises the functionality for obtaining instruction sets, data, and/or other information used, implemented, and/or executed by Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. DCADO Unit **100** comprises the functionality for modifying instruction sets, data, and/or other information used, implemented, and/or executed by Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. DCADO Unit **100** comprises learning, anticipating, decision making, automation, and/or other functionalities disclosed herein. Statistical, artificial intelligence, machine learning, and/or other models or techniques are utilized to implement the disclosed devices, systems, and methods.

When DCADO Unit **100** functionalities are applied on Application Program **18**, Processor **11**, Logic Circuit **250** (later described), and/or other processing element of Device **98**, Device **98** may become autonomous. DCADO Unit **100** may take control from, share control with, and/or release control to Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element to implement autonomous operation of Device **98**. DCADO Unit **100** may take control from, share control with, and/or release control to Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element automatically or after prompting User **50** to allow it. In some aspects, Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element of an autonomous Device **98** may include or be provided with anticipatory (also referred to as alternate or other suitable name or reference) instructions or instruction sets that User **50** did not issue or cause to be executed. Such anticipatory instructions or instruction sets include instruction sets that User **50** may want or is likely to issue or cause to be executed. Anticipatory instructions or instruction sets can be generated by DCADO Unit **100** or elements thereof based on Device's **98** circumstances including objects with various properties. As such, Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element of an autonomous Device **98** may include or be provided with some or all original instructions or instruction sets and/or any anticipatory instructions or instruction sets generated by DCADO Unit **100**. Therefore, autonomous Device **98** operating may include executing some or all original instructions or instruction sets and/or any anticipatory instructions or instruction sets generated by DCADO Unit **100**. In one example, DCADO Unit **100** can overwrite or rewrite the original instructions or instruction sets of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element with DCADO Unit **100**-generated instructions or instruction sets. In another example, DCADO Unit **100** can insert or embed DCADO Unit **100**-generated instructions or instruction sets among the original instructions or instruction sets of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. In a further example, DCADO Unit **100** can branch, redirect, or jump to DCADO Unit **100**-generated instructions or instruction sets from the original instructions or instruction sets of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element.

In some embodiments, autonomous Device **98** operating comprises determining, by DCADO Unit **100**, a next instruction or instruction set to be executed based on Device's **98** circumstances including objects with various properties prior to the user issuing or causing to be executed the next instruction or instruction set. In yet other embodiments, autonomous application operating comprises determining, by DCADO Unit **100**, a next instruction or instruction set to be executed based on Device's **98** circumstances including objects with various properties prior to the system receiving the next instruction or instruction set.

In some embodiments, autonomous Device **98** operating includes a partially or fully autonomous operating. In an example involving partially autonomous Device **98** operating, a user confirms DCADO Unit **100**-generated instructions or instruction sets prior to their execution. In an example involving fully autonomous application operating, DCADO Unit **100**-generated instructions or instruction sets are executed without user or other system confirmation (i.e. automatically, etc.).

In some embodiments, a combination of DCADO Unit **100** and other systems and/or techniques can be utilized to implement Device's **98** operation. In one example, DCADO Unit **100** may be a primary or preferred system for implementing Device's **98** operation. While operating autonomously under the control of DCADO Unit **100**, Device **98** may encounter a circumstance including objects with various properties that has not been encountered or learned before. In such situations, User **50** and/or non-DCADO system may take control of Device's **98** operation. DCADO Unit **100** may take control again when Device **98** encounters a previously learned circumstance including objects with various properties. Naturally, DCADO Unit **100** can learn Device's **98** operation in circumstances while User **50** and/or non-DCADO system is in control of Device **98**, thereby reducing or eliminating the need for future involvement of User **50** and/or non-DCADO system. In another example, User **50** and/or non-DCADO system may be a primary or preferred system for implementing Device's **98** operation. While operating under the control of User **50** and/or non-DCADO system, User **50** and/or non-DCADO system may release control to DCADO Unit **100** for any reason (i.e. User **50** gets tired or distracted, non-DCADO system gets stuck or cannot make a decision, etc.), at which point Device **98** can be controlled by DCADO Unit **100**. In some designs, DCADO Unit **100** may take control in certain special circumstances including objects with various properties where DCADO Unit **100** may offer superior performance even though User **50** and/or non-DCADO system may generally be preferred. Once Device **98** leaves such special circumstances, DCADO Unit **100** may release control to User **50** and/or non-DCADO system. In general, DCADO Unit **100** can take control from, share control with, or release control to User **50**, non-DCADO system, and/or other system or process at any time, in any circumstances, and remain in control for any period of time as needed.

In some embodiments, DCADO Unit **100** may control one or more sub-devices, sub-systems, or elements of Device **98** while User **50** and/or non-DCADO system may control other one or more sub-devices, sub-systems, or elements of Device **98**.

It should be understood that a reference to autonomous operating of Device **98** may include autonomous operating of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element depending on context.

Referring now to Acquisition Interface **120**, Acquisition Interface **120** comprises the functionality for obtaining and/or receiving instruction sets, data, and/or other information. Acquisition Interface **120** comprises the functionality for obtaining and/or receiving instruction sets, data, and/or other information from Processor **11**, Application Program **18**, Logic Circuit **250** (later described), and/or other processing element. Acquisition Interface **120** comprises the functionality for obtaining and/or receiving instruction sets, data, and/or other information at runtime. In some aspects, an instruction set may include any computer command, instruction, signal, or input used in Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. Therefore, the terms instruction set, command, instruction, signal, input, or other such terms may be used interchangeably herein depending on context. Acquisition Interface **120** also comprises the functionality for attaching to or interfacing with Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. In one example, Acquisition Interface **120** comprises the functionality to access and/or read runtime engine/environment, virtual machine, operating system, compiler, just-in-time (JIT) compiler, interpreter, translator, execution stack, file, object, data structure, and/or other computing system elements. In another example, Acquisition Interface **120** comprises the functionality to access and/or read memory, storage, bus, interfaces, and/or other computing system elements. In a further example, Acquisition Interface **120** comprises the functionality to access and/or read Processor **11** registers and/or other Processor **11** elements. In a further example, Acquisition Interface **120** comprises the functionality to access and/or read inputs and/or outputs of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. In a further example, Acquisition Interface **120** comprises the functionality to access and/or read functions, methods, procedures, routines, subroutines, and/or other elements of Application Program **18**. In a further example, Acquisition Interface **120** comprises the functionality to access and/or read source code, bytecode, compiled, interpreted, or otherwise translated code, machine code, and/or other code. In a further example, Acquisition Interface **120** comprises the functionality to access and/or read values, variables, parameters, and/or other data or information. Acquisition Interface **120** also comprises the functionality for transmitting the obtained instruction sets, data, and/or other information to Artificial Intelligence Unit **110** and/or other element. As such, Acquisition Interface **120** provides input into Artificial Intelligence Unit **110** for knowledge structuring, anticipating, decision making, and/or other functionalities later in the process. Acquisition Interface **120** also comprises other disclosed functionalities.

Acquisition Interface **120** can employ various techniques for obtaining instruction sets, data, and/or other information. In one example, Acquisition Interface **120** can attach to and/or obtain Processor's **11**, Application Program's **18**, Logic Circuit's **250**, and/or other processing element's instruction sets, data, and/or other information through tracing or profiling techniques. Tracing or profiling may be used for outputting Processor's **11**, Application Program's **18**, Logic Circuit's **250**, and/or other processing element's instruction sets, data, and/or other information at runtime. For instance, tracing or profiling may include adding trace code (i.e. instrumentation, etc.) to an application and/or outputting trace information to a specific target. The outputted trace information (i.e. instruction sets, data, and/or other information, etc.) can then be provided to or recorded into a file, data structure, repository, an application, and/or other system or target that may receive such trace information. As such, Acquisition Interface **120** can utilize tracing or profiling to obtain instruction sets, data, and/or other information and provide them as input into Artificial Intelligence Unit **110**. In some aspects, instrumentation can be performed in source code, bytecode, compiled, interpreted, or otherwise translated code, machine code, and/or other code. In other aspects, instrumentation can be performed in various elements of a computing system such as memory, virtual machine, runtime engine/environment, operating system, compiler, interpreter, translator, processor registers, execution stack, program counter, and/or other elements. In yet other aspects, instrumentation can be performed in various abstraction layers of a computing system such as in software layer (i.e. Application Program **18**, etc.), in virtual machine (if VM is used), in operating system, in Processor **11**, and/or in other layers or areas that may exist in a particular computing system implementation. In yet other aspects, instrumentation can be performed at various time periods in an application's execution such as source code write time, compile time, interpretation time, translation time, linking time, loading time, runtime, and/or other time periods. In yet other aspects, instrumentation can be performed at various granularities or code segments such as some or all lines of code, some or all statements, some or all instructions or instruction sets, some or all basic blocks, some or all functions/routines/subroutines, and/or some or all other code segments.

In some embodiments, Application Program **18** can be automatically instrumented. For example, Acquisition Interface **120** can access Application Program's **18** source code, bytecode, or machine code and select instrumentation points of interest. Selecting instrumentation points may include finding locations in the source code, bytecode, or machine code corresponding to function calls, function entries, function exits, object creations, object destructions, event handler calls, new lines (i.e. to instrument all lines of code, etc.), thread creations, throws, and/or other points of interest. Instrumentation code can then be inserted at the instrumentation points of interest to output Application Program's **18** instruction sets, data, and/or other information. In response to executing instrumentation code, Application Program's **18** instruction sets, data, and/or other information may be received by Acquisition Interface **120**. In some aspects, Application Program's **18** source code, bytecode, or machine code can be dynamically instrumented. For example, instrumentation code can be dynamically inserted into Application Program **18** at runtime.

In other embodiments, Application Program **18** can be manually instrumented. In one example, a programmer can instrument a function call by placing an instrumenting instruction immediately after the function call as in the following example.

```
Device1.moveForward(12);
traceApplication('Device1.moveForward(12);');
```

In another example, an instrumenting instruction can be placed immediately before the function call, or at the beginning, end, or anywhere within the function itself. A programmer may instrument all function calls or only function calls of interest. In a further example, a programmer can instrument all lines of code or only code lines of interest. In a further example, a programmer can instrument other elements utilized or implemented within Application Program **18** such as objects and/or any of their functions, data structures and/or any of their functions, event handlers and/or any of their functions, threads and/or any of their functions, and/or other elements or functions. Similar instru-

mentation as in the preceding examples can be performed automatically or dynamically. In some designs where manual code instrumentation is utilized, Acquisition Interface **120** can optionally be omitted and Application Program's **18** instruction sets, data, and/or other information may be transmitted directly to Artificial Intelligence Unit **110**.

In some embodiments, DCADO Unit **100** can be selective in learning instruction sets, data, and/or other information to those implemented, utilized, or related to an object, data structure, repository, thread, function, and/or other element of Application Program **18**. In some aspects, Acquisition Interface **120** can obtain Application Program's **18** instruction sets, data, and/or other information implemented, utilized, or related to a certain object in an object oriented Application Program **18**.

In some embodiments, various computing systems and/or platforms may provide native tools for obtaining instruction sets, data, and/or other information. Also, independent vendors may provide portable tools with similar functionalities that can be utilized across different computing systems and/or platforms. These native and portable tools may provide a wide range of functionalities to obtain runtime and other information such as instrumentation, tracing or profiling, logging application or system messages, outputting custom text messages, outputting objects or data structures, outputting functions/routines/subroutines or their invocations, outputting variable or parameter values, outputting thread or process behaviors, outputting call or other stacks, outputting processor registers, providing runtime memory access, providing inputs and/or outputs, performing live application monitoring, and/or other capabilities. One of ordinary skill in art will understand that, while all possible variations of the techniques to obtain instruction sets, data, and/or other information are too voluminous to describe, these techniques are within the scope of this disclosure.

In one example, obtaining instruction sets, data, and/or other information can be implemented through the .NET platform's native tools for application tracing or profiling such as System.Diagnostics.Trace, System.Diagnostics.Debug, and System.Diagnostics.TraceSource classes for tracing execution flow, and System. Diagnostics. Process, System.Diagnostics.EventLog, and System. Diagnostics. PerformanceCounter classes for profiling code, accessing local and remote processes, starting and stopping system processes, and interacting with Windows event logs, etc. For instance, a set of trace switches can be created that output an application's information. The switches can be configured using the .config file. For a Web application, this may typically be Web.config file associated with the project. In a Windows application, this file may typically be named applicationName.exe.config. Trace code can be added to application code automatically or manually as previously described. Appropriate listener can be created where the trace output is received. Trace code may output trace messages to a specific target such as a file, a log, a database, an object, a data structure, and/or other repository or system. Acquisition Interface **120** or Artificial Intelligence Unit **110** can then read or obtain the trace information from these targets. In some aspects, trace code may output trace messages directly to Acquisition Interface **120**. In other aspects, trace code may output trace messages directly to Artificial Intelligence Unit **110**. In the case of outputting trace messages to Acquisition Interface **120** or directly to Artificial Intelligence Unit **110**, custom listeners can be built to accommodate these specific targets. Other platforms, tools, and/or techniques can provide equivalent or similar functionalities as the above described ones.

In another example, obtaining instruction sets, data, and/or other information can be implemented through the .NET platform's Profiling API that can be used to create a custom profiler application for tracing, monitoring, interfacing with, and/or managing a profiled application. The Profiling API provides an interface that includes methods to notify the profiler of events in the profiled application. The Profiling API may also provide an interface to enable the profiler to call back into the profiled application to obtain information about the state of the profiled application. The Profiling API may further provide call stack profiling functionalities. Call stack (also referred to as execution stack, control stack, runtime stack, machine stack, the stack, etc.) includes a data structure that can store information about active subroutines of an application. The Profiling API may provide a stack snapshot method, which enables a trace of the stack at a particular point in time. The Profiling API may also provide a shadow stack method, which tracks the call stack at every instant. A shadow stack can obtain function arguments, return values, and information about generic instantiations. A function such as FunctionEnter can be utilized to notify the profiler that control is being passed to a function and can provide information about the stack frame and function arguments. A function such as FunctionLeave can be utilized to notify the profiler that a function is about to return to the caller and can provide information about the stack frame and function return value. An alternative to call stack profiling includes call stack sampling in which the profiler can periodically examine the stack. In some aspects, the Profiling API enables the profiler to change the in-memory code stream for a routine before it is just-in-time (JIT) compiled where the profiler can dynamically add instrumentation code to all or particular routines of interest. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through Java platform's APIs for application tracing or profiling such as Java Virtual Machine Profiling Interface (JVMPI), Java Virtual Machine Tool Interface (JVMTI), and/or other APIs or tools. These APIs can be used for instrumentation of an application, for notification of Java Virtual Machine (VM) events, and/or other functionalities. One of the tracing or profiling techniques that can be utilized includes bytecode instrumentation. The profiler can insert bytecodes into all or some of the classes. In application execution profiling, for example, these bytecodes may include methodEntry and methodExit calls. In memory profiling, for example, the bytecodes may be inserted after each new or after each constructor. In some aspects, insertion of instrumentation bytecode can be performed either by a post-compiler or a custom class loader. An alternative to bytecode instrumentation includes monitoring events generated by the JVMPI or JVMTI interfaces. Both APIs can generate events for method entry/exit, object allocation, and/or other events. In some aspects, JVMTI can be utilized for dynamic bytecode instrumentation where insertion of instrumentation bytecodes is performed at runtime. The profiler may insert the necessary instrumentation when a selected class is invoked in an application. This can be accomplished using the JVMTI's redefineClasses method, for example. This approach also enables changing of the level of profiling as the application is running. If needed, these changes can be made adaptively without restarting the application. Other

platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through JVM-TI's programming interface that enables creation of software agents that can monitor and control a Java application. An agent may use the functionality of the interface to register for notification of events as they occur in the application, and to query and control the application. A JVMTI agent may use JVMTI functions to extract information from a Java application. A JVMTI agent can be utilized to obtain an application's runtime information such as method calls, memory allocation, CPU utilization, lock contention, and/or other information. JVMTI may include functions to obtain information about variables, fields, methods, classes, and/or other information. JVMTI may also provide notification for numerous events such as method entry and exit, exception, field access and modification, thread start and end, and/or other events. Examples of JVMTI built-in methods include GetMethodName to obtain the name of an invoked method, GetThreadInfo to obtain information for a specific thread, GetClassSignature to obtain information about the class of an object, GetStackTrace to obtain information about the stack including information about stack frames, and/or other methods. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through java.lang.Runtime class that provides an interface for application tracing or profiling. Examples of methods provided in java.lang.Runtime that can be used to obtain an application's instruction sets, data, and/or other information include tracemethodcalls, traceinstructions, and/or other methods. These methods prompt the Java Virtual Machine to output trace information for a method or instruction in the virtual machine as it is executed. The destination of trace output may be system dependent and include a file, a listener, and/or other destinations where Acquisition Interface **120**, Artificial Intelligence Unit **110**, and/or other disclosed elements can access needed information. In addition to tracing or profiling tools native to their respective computing systems and/or platforms, many independent tools exist that provide tracing or profiling functionalities on more than one computing system and/or platform. Examples of these tools include Pin, DynamoRIO, KernInst, DynInst, Kprobes, OpenPAT, DTrace, SystemTap, and/or others. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through logging tools of the platform and/or operating system on which an application runs. Some logging tools may include nearly full feature sets of the tracing or profiling tools previously described. In one example, Visual Basic enables logging of runtime messages through its Microsoft.VisualBasic.Logging namespace that provides a log listener where the log listener may direct logging output to a file and/or other target. In another example, Java enables logging through its java.util.logging class. In some aspects, obtaining an application's instruction sets, data, and/or other information can be implemented through logging capabilities of the operating system on which an application runs. For example, Windows NT features centralized log service that applications and operating-system components can utilize to report their events including any messages. Windows NT provides functionalities for system, application, security, and/or other logging. An application log may include events logged by applications. Windows NT, for example, may include support for defining an event source (i.e. application that created the event, etc.). Windows Vista, for example, supports a structured XML log-format and designated log types to allow applications to more precisely log events and to help interpret the events. Examples of different types of event logs include administrative, operational, analytic, debug, and/or other log types including any of their subcategories. Examples of event attributes that can be utilized include eventID, level, task, opcode, keywords, and/or other event attributes. Windows wevtutil tool enables access to events, their structures, registered event publishers, and/or their configuration even before the events are fired. Wevtutil supports capabilities such as retrieval of the names of all logs on a computing device; retrieval of configuration information for a specific log; retrieval of event publishers on a computing device; reading events from an event log, from a log file, or using a structured query; exporting events from an event log, from a log file, or using a structured query to a specific target; and/or other capabilities. Operating system logs can be utilized solely if they contain sufficient information on an application's instruction sets, data, and/or other information. Alternatively, operating system logs can be utilized in combination with another source of information (i.e. trace information, call stack, processor registers, memory, etc.) to reconstruct the application's instruction sets, data, and/or other information needed for Artificial Intelligence Unit **110** and/or other elements. In addition to logging capabilities native to their respective platforms and/or operating systems, many independent tools exist that provide logging on different platforms and/or operating systems. Examples of these tools include Log4j, Logback, Smartinspect, NLog, log4net, Microsoft Enterprise Library, ObjectGuy Framework, and/or others. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through tracing or profiling the operating system on which an application runs. As in tracing or profiling an application, one of the techniques that can be utilized includes adding instrumentation code to the operating system's source code. Such instrumentation code can be added to the operating system's source code before kernel compilation or recompilation, for instance. This type of instrumentation may involve defining or finding locations in the operating system's source code where instrumentation code may be inserted. Kernel instrumentation can also be performed without the need for kernel recompilation or rebooting. In some aspects, instrumentation code can be added at locations of interest through binary rewriting of compiled kernel code. In other aspects, kernel instrumentation can be performed dynamically where instrumentation code is added and/or removed where needed at runtime. For instance, dynamic instrumentation may overwrite kernel code with a branch instruction that redirects execution to instrumentation code or instrumentation routine. In yet other aspects, kernel instrumentation can be performed using just-in-time (JIT) dynamic instrumentation where execution may be redirected to a copy of kernel's code segment that includes instrumentation code. This type of instrumentation may include a JIT compiler and creation of a copy of the original code segment having instrumentation code or calls to instrumentation routines embedded into the original code segment. Instrumentation of the operating system may enable total system visibility including visibility into an application's behavior by enabling genera-

tion of low level trace information. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through tracing or profiling the processor on which an application runs. For example, some Intel processors provide Intel Processor Trace (i.e. Intel PT, etc.), a low-level tracing feature that enables recording executed instruction sets, and/or other data or information of one or more applications. Intel PT is facilitated by the Processor Trace Decoder Library along with its related tools. Intel PT is a low-overhead execution tracing feature that records information about application execution on each hardware thread using dedicated hardware facilities. The recorded execution/trace information is collected in data packets that can be buffered internally before being sent to a memory subsystem or another system or element (i.e. Acquisition Interface **120**, Artificial Intelligence Unit **110**, etc.). Intel PT also enables navigating the recorded execution/trace information via reverse stepping commands. Intel PT can be included in an operating system's core files and provided as a feature of the operating system. Intel PT can trace globally some or all applications running on an operating system. Acquisition Interface **120** or Artificial Intelligence Unit **110** can read or obtain the recorded execution/trace information from Intel PT. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through branch tracing or profiling. Branch tracing may include an abbreviated instruction trace in which only the successful branch instruction sets are traced or recorded. Branch tracing can be implemented through utilizing dedicated processor commands, for example. Executed branches may be saved into special branch trace store area of memory. With the availability and reference to a compiler listing of the application together with branch trace information, a full path of executed instruction sets can be reconstructed. The full path can also be reconstructed with a memory dump (containing the program storage) and branch trace information. In some aspects, branch tracing can be utilized for pre-learning or automated learning of an application's instruction sets, data, and/or other information where a number of application simulations (i.e. simulations of likely/common operations, etc.) are performed. As such, the application's operation can be learned automatically saving the time that would be needed to learn the application's operation directed by a user. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through assembly language. Assembly language is a low-level programming language for a computer or other programmable device in which there is a strong correlation between the language and the architecture's machine instruction sets. Syntax, addressing modes, operands, and/or other elements of an assembly language instruction set may translate directly into numeric (i.e. binary, etc.) representations of that particular instruction set. Because of this direct relationship with the architecture's machine instruction sets, assembly language can be a powerful tool for tracing or profiling an application's execution in processor registers, memory, and/or other computing system components. For example, using assembly language, memory locations of a loaded application can be accessed, instrumented, and/or otherwise

manipulated. In some aspects, assembly language can be used to rewrite or overwrite original in-memory instruction sets of an application with instrumentation instruction sets. In other aspects, assembly language can be used to redirect application's execution to instrumentation routine/subroutine or other code segment elsewhere in memory by inserting a jump into the application's in-memory code, by redirecting program counter, or by other techniques. Some operating systems may implement protection from changes to applications loaded into memory. Operating system, processor, or other low level commands such as Linux mprotect command or similar commands in other operating systems may be used to unprotect the protected locations in memory before the change. In yet other aspects, assembly language can be used to obtain instruction sets, data, and/or other information through accessing and/or reading instruction register, program counter, other processor registers, memory locations, and/or other components of a computing system. In yet other aspects, high-level programming languages may call or execute an external assembly language program to facilitate obtaining instruction sets, data, and/or other information as previously described. In yet other aspects, relatively low-level programming languages such as C may allow embedding assembly language directly in their source code such as, for example, using asm keyword of C. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, it may be sufficient to obtain user or other inputs, variables, parameters, and/or other data in some procedural, simple object oriented, or other applications. In one instance, a simple procedural application executes a sequence of instruction sets until the end of the program. During its execution, the application may receive user or other input, store the input in a variable, and perform calculations using the variable to reach a result. The value of the variable can be obtained or traced. In another instance, a more complex procedural application comprises one or more functions/routines/subroutines each of which may include a sequence of instruction sets. The application may execute a main sequence of instruction sets with a branch to a function/routine/subroutine. During its execution, the application may receive user or other input, store the input in a variable, and pass the variable as a parameter to the function/routine/subroutine. The function/routine/subroutine may perform calculations using the parameter and return a value that the rest of the application can use to reach a result. The value of the variable or parameter passed to the function/routine/subroutine, and/or return value can be obtained or traced. Values of user or other inputs, variables, parameters, and/or other items of interest can be obtained through previously described tracing, instrumentation, and/or other techniques. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

Referring to FIG. **5**, in yet another example, obtaining instruction sets, data, and/or other information may be implemented through tracing, profiling, or sampling of instruction sets or data in processor registers, memory, or other computing system components where instruction sets, data, and/or other information may be stored or utilized. For example, Instruction Register **212** may be part of Processor **11** and it may store the instruction set currently being executed or decoded. In some processors, Program Counter **211** (also referred to as instruction pointer, instruction address register, instruction counter, or part of instruction sequencer) may be incremented after fetching an instruction set, and it may hold or point to the memory address of the

next instruction set to be executed. In a processor where the incrementation precedes the fetch, Program Counter **211** may point to the current instruction set being executed. In the instruction cycle, an instruction set may be loaded into Instruction Register **212** after Processor **11** fetches it from location in Memory **12** pointed to by Program Counter **211**. Instruction Register **212** may hold the instruction set while it is decoded by Instruction Decoder **213**, prepared, and executed. In some aspects, data (i.e. operands, etc.) needed for instruction set execution may be loaded from Memory **12** into a register within Register Array **214**. In other aspects, the data may be loaded directly into Arithmetic Logic Unit **215**. For instance, as instruction sets pass through Instruction Register **212** during application execution, they may be transmitted to Acquisition Interface **120** as shown. Examples of the steps in execution of a machine instruction set may include decoding the opcode (i.e. portion of a machine instruction set that may specify the operation to be performed), determining where the operands may be located (depending on architecture, operands may be in registers, the stack, memory, I/O ports, etc.), retrieving the operands, allocating processor resources to execute the instruction set (needed in some types of processors), performing the operation indicated by the instruction set, saving the results of execution, and/or other execution steps. Examples of the types of machine instruction sets that can be utilized include arithmetic, data handling, logical, program control, as well as special and/or other instruction set types. In addition to the ones described or shown, examples of other computing system or processor components that can be used during an instruction cycle include memory address register (MAR) that may hold the address of a memory block to be read from or written to; memory data register (MDR) that may hold data fetched from memory or data waiting to be stored in memory; data registers that may hold numeric values, characters, small bit arrays, or other data; address registers that may hold addresses used by instruction sets that indirectly access memory; general purpose registers (GPRs) that may store both data and addresses; conditional registers that may hold truth values often used to determine whether some instruction set should or should not be executed; floating point registers (FPRs) that may store floating point numbers; constant registers that may hold read-only values such as zero, one, or pi; special purpose registers (SPRs) such as status register, program counter, or stack pointer that may hold information on program state; machine-specific registers that may store data and settings related to a particular processor; Register Array **214** that may include an array of any number of processor registers; Arithmetic Logic Unit **215** that may perform arithmetic and logic operations; control unit that may direct processor's operation; and/or other circuits or components. Tracing, profiling, or sampling of processor registers, memory, or other computing system components can be implemented in a program, combination of hardware and program, or purely hardware system. Dedicated hardware may be built to perform tracing, profiling, or sampling of processor registers or any computing system components with marginal or no impact to computing overhead.

One of ordinary skill in art will recognize that FIG. **5** depicts one of many implementations of processor or computing system components, and that various additional components can be included, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate implementations. Processor or computing system components may be arranged or connected differently in alternate implementations. Processor or computing system components may also be connected with external elements using various connections. For instance, the connection between Instruction Register **212** and Acquisition Interface **120** may include any number or types of connections such as, for example, a dedicated connection for each bit of Instruction Register **212** (i.e. **32** connections for a 32 bit Instruction Register **212**, etc.). Any of the described or other connections or interfaces may be implemented among any processor or computing system components and Acquisition Interface **120** or other elements.

Referring to FIGS. **6**A-**6**B, in yet another example, obtaining instruction sets, data, and/or other information may be implemented through tracing, profiling, or sampling of Logic Circuit **250**. While Processor **11** includes any type or embodiment of logic circuit, Logic Circuit **250** is described separately here to offer additional detail on its functioning. Some Devices **98** may not need the processing capabilities of an entire Processor **11**, but instead a more tailored Logic Circuit **250**. Examples of such Devices **98** include home appliances, audio or video electronics, vehicle systems, toys, industrial machines, robots, and/or others. Logic Circuit **250** comprises the functionality for performing logic operations. Logic Circuit **250** comprises the functionality for performing logic operations using the circuit's inputs and producing outputs based on the logic operations performed on the inputs. Logic Circuit **250** may generally be implemented using transistors, diodes, and/or other electronic switches, but can also be constructed using vacuum tubes, electromagnetic relays (relay logic), fluidic logic, pneumatic logic, optics, molecules, or even mechanical elements. In some aspects, Logic Circuit **250** may be or include a microcontroller, field-programmable gate array (FPGA), application-specific integrated circuit (ASIC), and/or other computing circuit or device. In other aspects, Logic Circuit **250** may be or include any circuit or device comprising one or more logic gates, one or more transistors, one or more switches, and/or one or more other logic components. In further aspects, Logic Circuit **250** may be or include any integrated or other circuit or device that can perform logic operations. Logic may generally refer to Boolean logic utilized in binary operations, but other logics can also be used. Input into Logic Circuit **250** may include or refer to a value inputted into the Logic Circuit **250**, therefore, these terms may be used interchangeably herein depending on context. In one example, Logic Circuit **250** may perform some logic operations using four input values and produce two output values. As the four input values are delivered to or received by Logic Circuit **250**, they may be obtained by Acquisition Interface **120** through the four hardwired connections as shown in FIG. **6**A. In another example, Logic Circuit **250** may perform some logic operations using four input values and produce two output values. As the two output values are generated by or transmitted out of Logic Circuit **250**, they may be obtained by Acquisition Interface **120** through the two hardwired connections as shown in FIG. **6**B. In a further example, instead of or in addition to obtaining input and/or output values of Logic Circuit **250**, the state of Logic Circuit **250** may be obtained by reading or accessing values from one or more Logic Circuit's **250** internal components such as registers, memories, buses, and/or others (i.e. similar to the previously described tracing, profiling, and/or sampling of Processor **11** components, etc.). Tracing, profiling, or sampling of Logic Circuit **250** can be implemented in a program, combination of hardware and program, or purely hardware system. Dedicated hardware may be built to perform tracing, profiling, or sampling of Logic Circuit **250** with marginal or no impact to

computing overhead. Any of the elements and/or techniques for tracing, profiling, or sampling of Logic Circuit **250** can similarly be implemented with Processor **11** and/or other processing elements. In some designs, DCADO Unit **100** may include clamps and/or other elements to attach DCADO Unit **100** to inputs (i.e. input wires, etc.) into and/or outputs (i.e. output wires, etc.) from Logic Circuit **250**. Such clamps and/or attachment elements enable seamless attachment of DCADO Unit **100** to any circuit or computing device without the need to redesign or alter the circuit or computing device.

In some embodiments, DCADO Unit **100** may learn input values directly from an actuator (previously described, not shown). For example, Logic Circuit **250** or other processing element may control an actuator that enables Device **98** to perform mechanical, physical, and/or other operations. An actuator may receive one or more input values or control signals from Logic Circuit **250** or other processing element directing the actuator to perform specific operations. As one or more input values or control signals are delivered to or received by the actuator, they may be obtained by Acquisition Interface **120** as previously described with respect to obtaining input values of Logic Circuit **250**. Specifically, for instance, one or more input values or control signals of an actuator may be obtained by Acquisition Interface **120** via hardwired or other connections.

One of ordinary skill in art will recognize that FIGS. **6**A-**6**B depict one of many implementations of Logic Circuit **250** and that any number of input and/or output values can be utilized in alternate implementations. One of ordinary skill in art will also recognize that Logic Circuit **250** may include any number and/or combination of logic components to implement any logic operations.

Other additional techniques or elements may be utilized as needed for obtaining instruction sets, data, and/or other information, or some of the disclosed techniques or elements may be excluded, or a combination thereof may be utilized in alternate embodiments.

Referring to FIGS. **7**A-**7**E, some embodiments of Instruction Sets **526** are illustrated. In some aspects, Instruction Set **526** includes one or more instructions or commands of Application Program **18**. For example, Instruction Set **526** may include one or more instructions or commands of a high-level programming language such as Java or SQL, a low-level language such as assembly or machine language, an intermediate language or construct such as bytecode, and/or any other language or construct. In other aspects, Instruction Set **526** includes one or more inputs into and/or outputs from Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element. In further aspects, Instruction Set **526** includes one or more values or states of registers and/or other components of Logic Circuit **250**, Processor **11**, and/or other processing element. In general, Instruction Set **526** may include one or more instructions, commands, keywords, symbols (i.e. parentheses, brackets, commas, semicolons, etc.), operators (i.e. =, <, >, etc.), variables, values, objects, data structures, functions (i.e. Function1( ), FIRST( ) MIN( ) SQRT( ), etc.), parameters, states, signals, inputs, outputs, characters, digits, references thereto, and/or other components for performing an operation.

In an embodiment shown in FIG. **7**A, Instruction Set **526** includes code of a high-level programming language (i.e. Java, C++, etc.) comprising the following function call construct: Function1 (Parameter1, Parameter2, Parameter3, . . .). An example of a function call applying the above construct includes the following Instruction Set **526**: moveTo(Device1, 14, 8). The function or reference thereto "moveTo(Device1, 14, 8)" may be an Instruction Set **526** directing Device1 to move to a location with coordinates 14 and 8, for example. In another embodiment shown in FIG. **7**B, Instruction Set **526** includes structured query language (SQL). In a further embodiment shown in FIG. **7**C, Instruction Set **526** includes bytecode (i.e. Java bytecode, Python bytecode, CLR bytecode, etc.). In a further embodiment shown in FIG. **7**D, Instruction Set **526** includes assembly code. In a further embodiment shown in FIG. **7**E, Instruction Set **526** includes machine code.

Referring to FIGS. **8**A-**8**B, some embodiments of Extra Information **527** (also referred to as Extra Info **527**) are illustrated. In an embodiment shown in FIG. **8**A, Collection of Object Representations **525** may include or be associated with Extra Info **527**. In an embodiment shown in FIG. **8**B, Instruction Set **526** may include or be associated with Extra Info **527**.

Extra Info **527** comprises the functionality for storing any information useful in comparisons or decision making performed in autonomous device operation, and/or other functionalities. One or more Extra Infos **527** can be stored in, appended to, or associated with a Collection of Object Representations **525**, Instruction Set **526**, and/or other element. In some embodiments, the system can obtain Extra Info **527** at a time of creating or generating Collection of Object Representations **525**. In other embodiments, the system can obtain Extra Info **527** at a time of acquiring Instruction Set **526**. In general, Extra Info **527** can be obtained at any time. Examples of Extra Info **527** include time information, location information, computed information, contextual information, and/or other information. Any information can be utilized that can provide information for enhanced comparisons or decision making performed in autonomous device operation. Which information is utilized and/or stored in Extra Info **527** can be set by a user, by DCADO system administrator, or automatically by the system. Extra Info **527** may include or be referred to as contextual information, and vice versa. Therefore, these terms may be used interchangeably herein depending on context.

In some aspects, time information (i.e. time stamp, etc.) can be utilized and/or stored in Extra Info **527**. Time information can be useful in comparisons or decision making performed in autonomous device operation related to a specific time period as Device **98** may be required to perform specific operations at certain parts of day, month, year, and/or other time periods. Time information can be obtained from the system clock, online clock, oscillator, or other time source. In general, Extra Info **527** may include time information related to when Device **98** performed an operation. In other aspects, location information (i.e. coordinates, distance/angle from a known point, address, etc.) can be utilized and/or stored in Extra Info **527**. Location information can be useful in comparisons or decision making performed in autonomous device operation related to a specific place as Device **98** may be required to perform specific operations at certain places. Location information can be obtained from a positioning system (i.e. radio signal triangulation, GPS capabilities, etc.), sensors, and/or other location system. In general, Extra Info **527** may include location information related to where Device **98** performed an operation. In further aspects, computed information can be utilized and/or stored in Extra Info **527**. Computed information can be useful in comparisons or decision making performed in autonomous device operation where information can be calculated, inferred, or derived from other

available information. DCADO Unit **100** and/or other disclosed elements may include computational functionalities to create Extra Info **527** by performing calculations or inferences using other information. In one example, Device's **98** speed can be computed or estimated from Device's **98** location and/or time information. In another example, Device's **98** bearing (i.e. angle or direction of movement, etc.) can be computed or estimated from Device's **98** location information by utilizing Pythagorean theorem, trigonometry, and/or other theorems, formulas, or disciplines. In a further example, speeds, bearings, distances, and/or other properties of objects around Device **98** can similarly be computed or inferred using known information. In further aspects, observed information can be utilized and/or stored in Extra Info **527**. In further aspects, other information can be utilized and/or stored in Extra Info **527**. Examples of such other information include user specific information (i.e. skill level, age, gender, etc.), group user information (i.e. access level, etc.), version of Application Program **18**, type of Application Program **18**, type of Processor **11**, type of Logic Circuit **250**, type of Device **98**, and/or other information.

Referring to FIG. **9**, an embodiment where DCADO Unit **100** is part of or operating on Processor **11** is illustrated. In one example, DCADO Unit **100** may be a hardware element or circuit embedded or built into Processor **11**. In another example, DCADO Unit **100** may be a program operating on Processor **11**.

Referring to FIG. **10**, an embodiment where DCADO Unit **100** resides on Server **96** accessible over Network **95** is illustrated. Any number of Devices **98** may connect to such remote DCADO Unit **100** and the remote DCADO Unit **100** may learn their operations in circumstances including objects with various properties. In turn, any number of Devices **98** can utilize the remote DCADO Unit **100** for autonomous operation in circumstances including objects with various properties. A remote DCADO Unit **100** can be offered as a network service (i.e. online application, etc.). In some aspects, a remote DCADO Unit **100** (i.e. global DCADO Unit **100**, etc.) may reside on the Internet and be available to all the world's Devices **98** configured to transmit their operations in circumstances including objects with various properties and/or configured to utilize the remote DCADO Unit **100** for autonomous operation in circumstances including objects with various properties. For example, multiple operators (i.e. Users **50**, etc.) may operate their Devices **98** where the Devices **98** may be configured to transmit their operations in circumstances including objects with various properties to a remote DCADO Unit **100**. Such remote DCADO Unit **100** enables learning of the operators' collective knowledge of operating Device **98** in circumstances including objects with various properties. Server **96** may be or include any type or form of a remote computing device such as an application server, a network service server, a cloud server, a cloud, and/or other remote computing device. Server **96** may include any features, functionalities, and embodiments of the previously described Computing Device **70**. It should be understood that Server **96** does not have to be a separate computing device and that Server **96**, its elements, or its functionalities can be implemented on Device **98**. Network **95** may include various networks, connection types, protocols, interfaces, APIs, and/or other elements or techniques known in art all of which are within the scope of this disclosure. Any of the previously described networks, network or connection types, networking interfaces, and/or other networking elements or techniques can similarly be utilized. Any of the disclosed elements may reside on Server **96** in alternate implementations. In one example, Artificial Intelligence Unit **110** can reside on Server **96** and Acquisition Interface **120** and/or Modification Interface **130** can reside on Device **98**. In another example, Knowledgebase **530** can reside on Server **96** and the rest of the elements of DCADO Unit **100** can reside on Device **98**. Any other combination of local and remote elements can be implemented.

Referring to FIG. **11**, an embodiment of learning and/or using Remote Device's **97** circumstances for autonomous Remote Device **97** operation is illustrated. In such embodiments, in addition to providing input into Object Processing Unit **93** for learning functionalities herein, Sensor **92** (i.e. Camera **92***a*, Radar **92***d*, Sonar **92***e*, etc.) can provide input into Display **21** or other device for User's **50** perception of Remote Device's **97** surrounding. As User **50** operates Remote Device **97**, DCADO Unit **100** may learn Remote Device's **97** operation in circumstances including objects with various properties. Such embodiments can be utilized in any situation where one device controls (i.e. remote controls, etc.) another device, any situation where some or all of the processing is on one device and sensor capabilities are on another device, and/or other situations. In one example, a drone controlling device (i.e. Device **98**, etc.) may send control signals to operate a drone (i.e. Remote Device **97**, etc.) and receive information on the drone's surrounding from Sensor **92** on the drone. In another example, a robot controlling device (i.e. Device **98**, etc.) may send control signals to operate a robot (i.e. Remote Device **97**, etc.) and receive information on the robot's surrounding from Sensor **92** on the robot. Any of the disclosed elements in addition to Sensor **92** may reside on Remote Device **97** in alternate implementations.

Referring to FIG. **12**, an embodiment of Artificial Intelligence Unit **110** is illustrated. Artificial Intelligence Unit **110** comprises interconnected Knowledge Structuring Unit **520**, Knowledgebase **530**, Decision-making Unit **540**, and Confirmation Unit **550**. Other additional elements can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate embodiments.

Artificial Intelligence Unit **110** comprises the functionality for learning Device's **98** operation in circumstances including objects with various properties. Artificial Intelligence Unit **110** comprises the functionality for learning one or more collections of object representations correlated with any instruction sets, data, and/or other information. In some aspects, Artificial Intelligence Unit **110** comprises the functionality for learning one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**. In other aspects, Artificial Intelligence Unit **110** comprises the functionality for learning one or more Collections of Object Representations **525** some of which may not be correlated with any Instruction Sets **526** and/or Extra Info **527**. Further, Artificial Intelligence Unit **110** comprises the functionality for anticipating Device's **98** operation in circumstances including objects with various properties. Artificial Intelligence Unit **110** comprises the functionality for anticipating one or more instruction sets, data, and/or other information. Artificial Intelligence Unit **110** comprises the functionality for anticipating one or more Instruction Sets **526** based on one or more incoming Collections of Object Representations **525**. Artificial Intelligence Unit **110** comprises the functionality for anticipating one or more Instruction Sets **526** to be used or executed in Device's **98** autonomous operation. Artificial Intelligence Unit **110** also comprises other disclosed functionalities.

Knowledge Structuring Unit **520**, Knowledgebase **530**, and Decision-making Unit **540** are described later.

Confirmation Unit **550** comprises the functionality for confirming, modifying, evaluating (i.e. rating, etc.), and/or canceling one or more anticipatory Instruction Sets **526**, and/or other functionalities. Confirmation Unit **550** is an optional element that can be omitted depending on implementation. In some embodiments, Confirmation Unit **550** can serve as a means of confirming anticipatory Instruction Sets **526**. For example, Decision-making Unit **540** may determine one or more anticipatory Instruction Sets **526** and provide them to User **50** for confirmation. User **50** may be provided with an interface (i.e. graphical user interface, selectable list of anticipatory Instruction Sets **526**, etc.) to approve or confirm execution of the anticipatory Instruction Sets **526**. In some aspects, Confirmation Unit **550** can automate User **50** confirmation. In one example, if one or more incoming Collections of Object Representations **525** from Object Processing Unit **93** and one or more Collections of Object Representations **525** from a Knowledge Cell **800** were found to be a perfect or highly similar match, anticipatory Instruction Sets **526** correlated with the one or more Collections of Object Representations **525** from the Knowledge Cell **800** can be automatically executed without User's **50** confirmation. Conversely, if one or more incoming Collections of Object Representations **525** from Object Processing Unit **93** and one or more Collections of Object Representations **525** from a Knowledge Cell **800** were found to be less than a highly similar match, anticipatory Instruction Sets **526** correlated with the one or more Collections of Object Representations **525** from the Knowledge Cell **800** can be presented to User **50** for confirmation and/or modifying. Any features, functionalities, and/or embodiments of Similarity Comparison **125** (later described) can be utilized for such similarity determination. In other embodiments, Confirmation Unit **550** can serve as a means of modifying or editing anticipatory Instruction Sets **526**. For example, Decision-making Unit **540** may determine one or more anticipatory Instruction Sets **526** and provide them to User **50** for modification. User **50** may be provided with an interface (i.e. graphical user interface, etc.) to modify the anticipatory Instruction Sets **526** before their execution. In further embodiments, Confirmation Unit **550** can serve as a means of evaluating or rating anticipatory Instruction Sets **526** if they matched User's **50** intended operation. For example, Decision-making Unit **540** may determine one or more anticipatory Instruction Sets **526**, which the system may automatically execute. User **50** may be provided with an interface (i.e. graphical user interface, etc.) to rate (i.e. on a scale from 0 to 1, etc.) how well Decision-making Unit **540** predicted the executed anticipatory Instruction Sets **526**. In some aspects, rating can be automatic and based on a particular function or method that rates how well the anticipatory Instruction Sets **526** matched the desired operation. In one example, a rating function or method can assign a higher rating to anticipatory Instruction Sets **526** that were least modified in the confirmation process. In another example, a rating function or method can assign a higher rating to anticipatory Instruction Sets **526** that were canceled least number of times by User **50**. Any other automatic rating function or method can be utilized. In yet other embodiments, Confirmation Unit **550** can serve as a means of canceling anticipatory Instruction Sets **526** if they did not match User's **50** intended operation. For example, Decision-making Unit **540** may determine one or more anticipatory Instruction Sets **526**, which the system may automatically execute. The system may save the state of Device **98**, Processor **11** (save its register values, etc.), Logic Circuit **250**, Application Program **18** (i.e. save its variables, data structures, objects, location of its current instruction, etc.), and/or other processing elements before executing anticipatory Instruction Sets **526**. User **50** may be provided with an interface (i.e. graphical user interface, selectable list of prior executed anticipatory Instruction Sets **526**, etc.) to cancel one or more of the prior executed anticipatory Instruction Sets **526**, and restore Device **98**, Processor **11**, Logic Circuit **250**, Application Program **18**, and/or other processing elements to a prior state. In some aspects, Confirmation Unit **550** can optionally be disabled or omitted in order to provide an uninterrupted operation of Device **98**, Processor **11**, Logic Circuit **250**, and/or Application Program **18**. For example, a thermostat may be suitable for implementing the user confirmation step, whereas, a vehicle may be less suitable for implementing such interrupting step due to the real time nature of vehicle operation.

Referring to FIG. **13**, an embodiment of Knowledge Structuring Unit **520** correlating individual Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527** is illustrated. Knowledge Structuring Unit **520** comprises the functionality for structuring the knowledge of a device's operation in circumstances including objects with various properties, and/or other functionalities. Knowledge Structuring Unit **520** comprises the functionality for correlating one or more Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527**. Knowledge Structuring Unit **520** comprises the functionality for creating or generating Knowledge Cell **800** and storing one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** into the Knowledge Cell **800**. As such, Knowledge Cell **800** comprises the functionality for storing one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**. Knowledge Cell **800** includes knowledge (i.e. unit of knowledge, etc.) of how Device **98** operated in a circumstance including objects with various properties. Once created or generated, Knowledge Cells **800** can be used in/as neurons, nodes, vertices, or other elements in any of the data structures or arrangements (i.e. neural networks, graphs, sequences, etc.) used for storing the knowledge of Device's **98** operation in circumstances including objects with various properties, thereby facilitating learning functionalities herein. It should be noted that Extra Info **527** may be optionally used in some implementations to enable enhanced comparisons or decision making in autonomous device operation where applicable, and that Extra Info **527** can be omitted in alternate implementations.

In some embodiments, Knowledge Structuring Unit **520** receives one or more Collections of Object Representations **525** from Object Processing Unit **93**. Knowledge Structuring Unit **520** may also receive one or more Instruction Sets **526** from Acquisition Interface **120**. Knowledge Structuring Unit **520** may further receive any Extra Info **527**. Although, Extra Info **527** is not shown in this and/or other figures for clarity of illustration, it should be noted that any Collection of Object Representations **525**, Instruction Set **526**, and/or other element may include or be associated with Extra Info **527**. Knowledge Structuring Unit **520** may correlate one or more Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527**. Knowledge Structuring Unit **520** may then create Knowledge Cell **800** and store the one or more Collections of Object Representations **525** correlated with Instruction Sets **526** and/or Extra Info **527** into the Knowledge Cell **800**. Knowledge Cell **800**

may include any data structure or arrangement that can facilitate such storing. For example, Knowledge Structuring Unit **520** may create Knowledge Cell **800***ax* and structure within it Collection of Object Representations **525***a***1** correlated with Instruction Sets **526***a***1**-**526***a***3** and/or any Extra Info **527** (not shown). Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800***ax* a Collection of Object Representations **525***a***2** correlated with Instruction Set **526***a***4** and/or any Extra Info **527** (not shown). Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800***ax* a Collection of Object Representations **525***a***3** without a correlated Instruction Set **526** and/or Extra Info **527**. Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800***ax* a Collection of Object Representations **525***a***4** correlated with Instruction Sets **526***a***5**-**526***a***6** and/or any Extra Info **527** (not shown). Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800***ax* a Collection of Object Representations **525***a***5** without a correlated Instruction Set **526** and/or Extra Info **527**. Knowledge Structuring Unit **520** may structure within Knowledge Cell **800***ax* additional Collections of Object Representations **525** correlated with any number (including zero [i.e. uncorrelated]) of Instruction Sets **526** and/or Extra Info **527** by following similar logic as described above.

In some embodiments, Knowledge Structuring Unit **520** may correlate a Collection of Object Representations **525** with one or more temporally corresponding Instruction Sets **526** and/or Extra Info **527**. This way, Knowledge Structuring Unit **520** can structure the knowledge of Device's **98** operation at or around the time of generating Collections of Object Representations **525**. Such functionality enables spontaneous or seamless learning of Device's **98** operation in circumstances including objects with various properties as Device **98** is operated in real life situations. In some designs, Knowledge Structuring Unit **520** may receive a stream of Instruction Sets **526** used or executed to effect Device's **98** operations as well as a stream of Collections of Object Representations **525** as the operations are performed. Knowledge Structuring Unit **520** can then correlate Collections of Object Representations **525** from the stream of Collections of Object Representations **525** with temporally corresponding Instruction Sets **526** from the stream of Instruction Sets **526** and/or any Extra Info **527**. Collections of Object Representations **525** without a temporally corresponding Instruction Set **526** may be uncorrelated, for instance. In some aspects, Instruction Sets **526** and/or Extra Info **527** that temporally correspond to a Collection of Object Representations **525** may include Instruction Sets **526** used and/or Extra Info **527** obtained at the time of generating the Collection of Object Representations **525**. In other aspects, Instruction Sets **526** and/or Extra Info **527** that temporally correspond to a Collection of Object Representations **525** may include Instruction Sets **526** used and/or Extra Info **527** obtained within a certain time period before and/or after generating the Collection of Object Representations **525**. For example, Instruction Sets **526** and/or Extra Info **527** that temporally correspond to a Collection of Object Representations **525** may include Instruction Sets **526** used and/or Extra Info **527** obtained within 50 milliseconds, 1 second, 3 seconds, 20 seconds, 1 minute, 41 minutes, 2 hours, or any other time period before and/or after generating the Collection of Object Representations **525**. Such time periods can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. In other aspects, Instruction Sets **526** and/or Extra Info **527** that temporally correspond to a Collection of Object Representations **525** may include Instruction Sets **526** used and/or Extra Info **527** obtained from the time of generating the Collection of Object Representations **525** to the time of generating a next Collection of Object Representations **525**. In further aspects, Instruction Sets **526** and/or Extra Info **527** that temporally correspond to a Collection of Object Representations **525** may include Instruction Sets **526** used and/or Extra Info **527** obtained from the time of generating a previous Collection of Object Representations **525** to the time of generating the Collection of Object Representations **525**. Any other temporal relationship or correspondence between Collections of Object Representations **525** and correlated Instruction Sets **526** and/or Extra Info **527** can be implemented.

In some embodiments, Knowledge Structuring Unit **520** can structure the knowledge of Device's **98** operation in a circumstance including objects with various properties into any number of Knowledge Cells **800**. In some aspects, Knowledge Structuring Unit **520** can structure into a Knowledge Cell **800** a single Collection of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**. In other aspects, Knowledge Structuring Unit **520** can structure into a Knowledge Cell **800** any number (i.e. **2**, **4**, **7**, **17**, **29**, **87**, **1415**, **23891**, **323674**, **8132401**, etc.) of Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**. In a special case, Knowledge Structuring Unit **520** can structure all Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** into a single long Knowledge Cell **800**. In further aspects, Knowledge Structuring Unit **520** can structure Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** into a plurality of Knowledge Cells **800**. In a special case, Knowledge Structuring Unit **520** can store periodic streams of Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** into a plurality of Knowledge Cells **800** such as hourly, daily, weekly, monthly, yearly, or other periodic Knowledge Cells **800**.

In some embodiments, Device **98** may include a plurality of Sensors **92** and/or their corresponding Object Processing Units **93**. In one example, multiple Sensors **92** may detect objects and/or their properties from different angles or on different sides of Device **98**. In another example, one or more Sensors **92** may be placed on different sub-devices, sub-systems, or elements of Device **98**. Using multiple Sensors **92** and/or their corresponding Object Processing Units **93** may provide additional detail in learning and/or using Device's **98** circumstances for autonomous Device **98** operation. In some designs where multiple Sensors **92** and/or their corresponding Object Processing Units **93** are utilized, multiple DCADO Units **100** can also be utilized (i.e. one DCADO Unit **100** for each Sensor **92** and its corresponding Object Processing Unit **93**, etc.). In such designs, Collections of Object Representations **525** can be correlated with any Instruction Sets **526** and/or Extra Info **527** as previously described. In other designs where multiple Sensors **92** and/or their corresponding Object Processing Units **93** are utilized, collective Collections of Object Representations **525** from multiple Sensors **92** and their corresponding Object Processing Units **93** can be correlated with any Instruction Sets **526** and/or Extra Info **527**.

In some embodiments, Device **98** may include a plurality of Logic Circuits **250**, Processors **11**, Application Programs **18**, and/or other processing elements. For example, each processing element may control a sub-device, sub-system, or

an element of Device **98**. Using multiple processing elements may provide enhanced control over Device's **98** operation. In some designs where multiple processing elements are utilized, multiple DCADO Units **100** can also be utilized (i.e. one DCADO Unit **100** for each processing element, etc.). In such designs, Collections of Object Representations **525** can be correlated with any Instruction Sets **526** and/or Extra Info **527** as previously described. In other designs where multiple processing elements are utilized, Collections of Object Representations **525** can be correlated with any collective Instruction Sets **526** and/or Extra Info **527** used or executed by a plurality of processing elements.

Any combination of the aforementioned multiple Sensors **92** and/or their corresponding Object Processing Units **93**, multiple processing elements, and/or other elements can be implemented in alternate embodiments.

Referring to FIG. **14**, another embodiment of Knowledge Structuring Unit **520** correlating individual Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527** is illustrated. In such embodiments, Knowledge Structuring Unit **520** may generate Knowledge Cells **800** each comprising a single Collection of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**.

Referring to FIG. **15**, an embodiment of Knowledge Structuring Unit **520** correlating streams of Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527** is illustrated. For example, Knowledge Structuring Unit **520** may create Knowledge Cell **800***ax* and structure within it a stream of Collections of Object Representations **525***a***1**-**525***an* correlated with Instruction Set **526***a***1** and/or any Extra Info **527** (not shown). Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800***ax* a stream of Collections of Object Representations **525***b***1**-**525***bn* correlated with Instruction Sets **526***a***2**-**526***a***4** and/or and Extra Info **527** (not shown). Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800***ax* a stream of Collections of Object Representations **525***c***1**-**525***cn* without correlated Instruction Sets **526** and/or Extra Info **527**. Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800***ax* a stream of Collections of Object Representations **525***d***1**-**525***dn* correlated with Instruction Sets **526***a***5**-**526***a***6** and/or any Extra Info **527** (not shown). Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800***ax* additional streams of Collections of Object Representations **525** correlated with any number (including zero [i.e. uncorrelated]) of Instruction Sets **526** and/or Extra Info **527** by following similar logic as described above. The number of Collections of Object Representations **525** in some or all streams of Collections of Object Representations **525***a***1**-**525***an*, **525***b***1**-**525***bn*, etc. may be equal or different. It should be noted that n or other such letters or indicia may follow the sequence and/or context where they are indicated. Also, a same letter or indicia such as n may represent a different number in different elements of a drawing.

Referring to FIG. **16**, another embodiment of Knowledge Structuring Unit **520** correlating streams of Collections of Object Representations **525** with any Instruction Sets **526** and/or Extra Info **527** is illustrated. In such embodiments, Knowledge Structuring Unit **520** may generate Knowledge Cells **800** each comprising a single stream of Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**.

Knowledgebase **530** comprises the functionality for storing the knowledge of a device's operation in circumstances including objects with various properties, and/or other functionalities. Knowledgebase **530** comprises the functionality for storing one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**. Knowledgebase **530** comprises the functionality for storing one or more Knowledge Cells **800** each including one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**. In some aspects, Collections of Object Representations **525** correlated with Instruction Sets **526** and/or Extra Info **527** can be stored directly within Knowledgebase **530** without using Knowledge Cells **800** as the intermediary data structures. In some embodiments, Knowledgebase **530** may be or include Neural Network **530***a* (later described). In other embodiments, Knowledgebase **530** may be or include Graph **530***b* (later described). In further embodiments, Knowledgebase **530** may be or include Collection of Sequences **530***c* (later described). In further embodiments, Knowledgebase **530** may be or include Sequence **533** (later described). In further embodiments, Knowledgebase **530** may be or include Collection of Knowledge Cells **530***d* (later described). In general, Knowledgebase **530** may be or include any data structure or arrangement capable of storing the knowledge of a device's operation in circumstances including objects with various properties. Knowledgebase **530** may reside locally on Device **98**, or remotely (i.e. remote Knowledgebase **530**, etc.) on a remote computing device (i.e. server, cloud, etc.) accessible over a network or an interface.

In some embodiments, Knowledgebase **530** from one Device **98** or DCADO Unit **100** can be transferred to one or more other Devices **98** or DCADO Units **100**. Therefore, the knowledge of Device's **98** operation in circumstances including objects with various properties learned on one Device **98** or DCADO Unit **100** can be transferred to one or more other Devices **98** or DCADO Units **100**. In one example, Knowledgebase **530** can be copied or downloaded to a file or other repository from one Device **98** or DCADO Unit **100** and loaded or inserted into another Device **98** or DCADO Unit **100**. In another example, Knowledgebase **530** from one Device **98** or DCADO Unit **100** can be available on a server accessible by other Devices **98** or DCADO Units **100** over a network or an interface. Once loaded into or accessed by a receiving Device **98** or DCADO Unit **100**, the receiving Device **98** or DCADO Unit **100** can then implement the knowledge of Device's **98** operation in circumstances including objects with various properties learned on the originating Device **98** or DCADO Unit **100**.

In some embodiments, multiple Knowledgebases **530** (i.e. Knowledgebases **530** from different Devices **98** or DCADO Units **100**, etc.) can be combined to accumulate collective knowledge of operating Device **98** in circumstances including objects with various properties. In one example, one Knowledgebase **530** can be appended to another Knowledgebase **530** such as appending one Collection of Sequences **530***c* (later described) to another Collection of Sequences **530***c*, appending one Sequence **533** (later described) to another Sequence **533**, appending one Collection of Knowledge Cells **530***d* (later described) to another Collection of Knowledge Cells **530***d*, and/or appending other data structures or elements thereof. In another example, one Knowledgebase **530** can be copied into another Knowledgebase **530** such as copying one Collection of Sequences **530***c* into another Collection of Sequences **530***c*, copying one Collection of Knowledge Cells **530***d* into another Collection of Knowledge Cells **530***d*, and/or copying other data structures or elements thereof. In a further

example, in the case of Knowledgebase **530** being or including Graph **530***b* or graph-like data structure (i.e. Neural Network **530***a*, tree, etc.), a union can be utilized to combine two or more Graphs **530***b* or graph-like data structures. For instance, a union of two Graphs **530***b* or graph-like data structures may include a union of their vertex (i.e. node, etc.) sets and their edge (i.e. connection, etc.) sets. Any other operations or combination thereof on graphs or graph-like data structures can be utilized to combine Graphs **530***b* or graph-like data structures. In a further example, one Knowledgebase **530** can be combined with another Knowledgebase **530** through later described learning processes where Knowledge Cells **800** may be applied one at a time and connected with prior and/or subsequent Knowledge Cells **800** such as in Graph **530***b* or Neural Network **530***a*. In such embodiments, instead of Knowledge Cells **800** generated by Knowledge Structuring Unit **520**, the learning process may utilize Knowledge Cells **800** from one Knowledgebase **530** to apply them onto another Knowledgebase **530**. Any other techniques known in art including custom techniques for combining data structures can be utilized for combining Knowledgebases **530** in alternate implementations. In any of the aforementioned and/or other combining techniques, similarity of elements (i.e. nodes/vertices, edges/connections, etc.) can be utilized in determining whether an element from one Knowledgebase **530** matches an element from another Knowledgebase **530**, and substantially or otherwise similar elements may be considered a match for combining purposes in some designs. Any features, functionalities, and embodiments of Similarity Comparison **125** (later described) can be used in such similarity determinations. A combined Knowledgebase **530** can be offered as a network service (i.e. online application, etc.), downloadable file, or other repository to all DCADO Units **100** configured to utilize the combined Knowledgebase **530**. For example, a Device **98** including or interfaced with DCADO Unit **100** having access to a combined Knowledgebase **530** can use the collective knowledge learned from multiple Devices **98** for the Device's **98** autonomous operation.

Referring to FIG. **17**, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation may include various artificial intelligence models and/or techniques. The disclosed systems, devices, and methods are independent of the artificial intelligence model and/or technique used and any model and/or technique can be utilized to facilitate the functionalities described herein. Examples of these models and/or techniques include deep learning, supervised learning, unsupervised learning, neural networks (i.e. convolutional neural network, recurrent neural network, deep neural network, etc.), search-based, logic and/or fuzzy logic-based, optimization-based, tree/graph/other data structure-based, hierarchical, symbolic and/or sub-symbolic, evolutionary, genetic, multi-agent, deterministic, probabilistic, statistical, and/or other models and/or techniques.

In one example shown in Model A, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation may include a neural network (also referred to as artificial neural network, etc.). As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include a network of Nodes **852** (also referred to as neurons, etc.) and Connections **853** similar to that of a brain. Node **852** can store any data, object, data structure, and/or other item, or reference thereto. Node **852** may also include a function for transforming or

manipulating any data, object, data structure, and/or other item. Examples of such transformation functions include mathematical functions (i.e. addition, subtraction, multiplication, division, sin, cos, log, derivative, integral, etc.), object manipulation functions (i.e. creating an object, modifying an object, deleting an object, appending objects, etc.), data structure manipulation functions (i.e. creating a data structure, modifying a data structure, deleting a data structure, creating a data field, modifying a data field, deleting a data field, etc.), and/or other transformation functions. Connection **853** may include or be associated with a value such as a symbolic label or numeric attribute (i.e. weight, cost, capacity, length, etc.). A computational model can be utilized to compute values from inputs based on a pre-programmed or learned function or method. For example, a neural network may include one or more input neurons that can be activated by inputs. Activations of these neurons can then be passed on, weighted, and transformed by a function to other neurons. Neural networks may range from those with only one layer of single direction logic to multi-layer of multi-directional feedback loops. A neural network can use weights to change the parameters of the network's throughput. A neural network can learn by input from its environment or from self-teaching using written-in rules. A neural network can be utilized as a predictive modeling approach in machine learning. An exemplary embodiment of a neural network (i.e. Neural Network **530***a*, etc.) is described later.

In another example shown in Model B, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation may include a graph or graph-like data structure. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include Nodes **852** (also referred to as vertices or points, etc.) and Connections **853** (also referred to as edges, arrows, lines, arcs, etc.) organized as a graph. In general, any Node **852** in a graph can be connected to any other Node **852**. A Connection **853** may include unordered pair of Nodes **852** in an undirected graph or ordered pair of Nodes **852** in a directed graph. Nodes **852** can be part of the graph structure or external entities represented by indices or references. A graph can be utilized as a predictive modeling approach in machine learning. Nodes **852**, Connections **853**, and/or other elements or operations of a graph may include any features, functionalities, and embodiments of the aforementioned Nodes **852**, Connections **853**, and/or other elements or operations of a neural network, and vice versa. An exemplary embodiment of a graph (i.e. Graph **530***b*, etc.) is described later.

In a further example shown in Model C, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation may include a tree or tree-like data structure. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include Nodes **852** and Connections **853** (also referred to as references, edges, etc.) organized as a tree. In general, a Node **852** in a tree can be connected to any number (i.e. including zero, etc.) of children Nodes **852**. A tree can be utilized as a predictive modeling approach in machine learning. Nodes **852**, Connections **853**, and/or other elements or operations of a tree may include any features, functionalities, and embodiments of the aforementioned Nodes **852**, Connections **853**, and/or other elements or operations of a neural network and/or graph, and vice versa.

In a further example shown in Model D, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation may include a sequence or sequence-like data structure. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include a structure of Nodes **852** and/or Connections **853** organized as a sequence. In some aspects, Connections **853** may be optionally omitted from a sequence as the sequential order of Nodes **852** in a sequence may be implied in the structure. A sequence can be utilized as a predictive modeling approach in machine learning. Nodes **852**, Connections **853**, and/or other elements or operations of a sequence may include any features, functionalities, and embodiments of the aforementioned Nodes **852**, Connections **853**, and/or other elements or operations of a neural network, graph, and/or tree, and vice versa. An exemplary embodiment of a sequence (i.e. Collection of Sequences **530***c*, Sequence **533**, etc.) is described later.

In yet another example, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation may include a search-based model and/or technique. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include searching through a collection of possible solutions. For example, a search method can search through a neural network, graph, tree, sequence, or other data structure that includes data elements of interest. A search may use heuristics to limit the search for solutions by eliminating choices that are unlikely to lead to the goal. Heuristic techniques may provide a best guess solution. A search can also include optimization. For example, a search may begin with a guess and then refine the guess incrementally until no more refinements can be made. In a further example, the disclosed systems, devices, and methods may include logic-based model and/or technique. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities can use formal or other type of logic. Logic based models may involve making inferences or deriving conclusions from a set of premises. As such, a logic based system can extend existing knowledge or create new knowledge automatically using inferences. Examples of the types of logic that can be utilized include propositional or sentential logic that comprises logic of statements which can be true or false; first-order logic that allows the use of quantifiers and predicates and that can express facts about objects, their properties, and their relations with each other; fuzzy logic that allows degrees of truth to be represented as a value between 0 and 1 rather than simply 0 (false) or 1 (true), which can be used for uncertain reasoning; subjective logic that comprises a type of probabilistic logic that may take uncertainty and belief into account, which can be suitable for modeling and analyzing situations involving uncertainty, incomplete knowledge and different world views; and/or other types of logic. In a further example, the disclosed systems, devices, and methods may include a probabilistic model and/or technique. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities can be implemented to operate with incomplete or uncertain information where probabilities may affect outcomes. Bayesian network, among other models, is an example of a probabilistic tool used for purposes such as reasoning, learning, planning, perception, and/or others. One of ordinary skill in art will understand that the aforementioned artificial intelligence models and/or techniques are described merely as examples of a variety of possible implementations, and that while all possible artificial intelligence models and/or techniques are too voluminous to describe, other artificial intelligence models and/or techniques known in art are within the scope of this disclosure. One of ordinary skill in art will also recognize that an intelligent system may solve a specific problem by using any model and/or technique that works such as, for example, some systems can be symbolic and logical, some can be sub-symbolic neural networks, some can be deterministic or probabilistic, some can be hierarchical, some may include searching techniques, some may include optimization techniques, while others may use other or a combination of models and/or techniques. In general, any artificial intelligence model and/or technique can be utilized that can facilitate the functionalities described herein.

Referring to FIGS. **18**A-**18**C, embodiments of interconnected Knowledge Cells **800** and updating weights of Connections **853** are illustrated. As shown for example in FIG. **18**A, Knowledge Cell **800***za* is connected to Knowledge Cell **800***zb* and Knowledge Cell **800***zc* by Connection **853***z***1** and Connection **853***z***2**, respectively. Each of Connection **853***z***1** and Connection **853***z***2** may include or be associated with occurrence count, weight, and/or other parameter or data. The number of occurrences may track or store the number of observations that a Knowledge Cell **800** was followed by another Knowledge Cell **800** indicating a connection or relationship between them. For example, Knowledge Cell **800***za* was followed by Knowledge Cell **800***zb* 10 times as indicated by the number of occurrences of Connection **853***z***1**. Also, Knowledge Cell **800***za* was followed by Knowledge Cell **800***zc* 15 times as indicated by the number of occurrences of Connection **853***z***2**. The weight of Connection **853***z***1** can be calculated or determined as the number of occurrences of Connection **853***z***1** divided by the sum of occurrences of all connections (i.e. Connection **853***z***1** and Connection **853***z***2**, etc.) originating from Knowledge Cell **800***za*. Therefore, the weight of Connection **853***z***1** can be calculated or determined as 10/(10+15)=0.4, for example. Also, the weight of Connection **853***z***2** can be calculated or determined as 15/(10+15)=0.6, for example. Therefore, the sum of weights of Connection **853***z***1**, Connection **853***z***2**, and/or any other Connections **853** originating from Knowledge Cell **800***za* may equal to 1 or 100%. As shown for example in FIG. **18**B, in the case that Knowledge Cell **800***zd* is inserted and an observation is made that Knowledge Cell **800***zd* follows Knowledge Cell **800***za*, Connection **853***z***3** can be created between Knowledge Cell **800***za* and Knowledge Cell **800***zd*. The occurrence count of Connection **853***z***3** can be set to 1 and weight determined as 1/(10+15+1)=0.038. The weights of all other connections (i.e. Connection **853***z***1**, Connection **853***z***2**, etc.) originating from Knowledge Cell **800***za* may be updated to account for the creation of Connection **853***z***3**. Therefore, the weight of Connection **853***z***1** can be updated as 10/(10+15-1)=0.385. The weight of Connection **853***z***2** can also be updated as 15/(10+15+1)=0.577. As shown for example in FIG. **18**C, in the case that an additional occurrence of Connection **853***z***1** is observed (i.e. Knowledge Cell **800***zb* followed Knowledge Cell **800***za*, etc.), occurrence count of Connection **853***z***1** and weights of all connections (i.e. Connection **853***z***1**, Connection **853***z***2**, and Connection **853***z***3**, etc.) originating from Knowledge Cell **800***za* may be updated to account for this observation. The occurrence count of Connection **853***z***1**

can be increased by 1 and its weight updated as 11/(11+15+1)=0.407. The weight of Connection **853**z**2** can also be updated as 15/(11+15+1)=0.556. The weight of Connection **853**z**3** can also be updated as 1/(11+15+1)=0.037.

Referring to FIG. **19**, an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Collection of Knowledge Cells **530***d* is illustrated. Collection of Knowledge Cells **530***d* comprises the functionality for storing any number of Knowledge Cells **800**. In some aspects, Knowledge Cells **800** may be stored into or applied onto Collection of Knowledge Cells **530***d* in a learning or training process. In effect, Collection of Knowledge Cells **530***d* may store Knowledge Cells **800** that can later be used to enable autonomous Device **98** operation. In some embodiments, Knowledge Structuring Unit **520** structures or generates Knowledge Cells **800** as previously described and the system applies them onto Collection of Knowledge Cells **530***d*, thereby implementing learning Device's **98** operation in circumstances including objects with various properties. The term apply or applying may refer to storing, copying, inserting, updating, or other similar action, therefore, these terms may be used interchangeably herein depending on context. The system can perform Similarity Comparisons **125** (later described) of a newly structured Knowledge Cell **800** from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. If a substantially similar Knowledge Cell **800** is not found in Collection of Knowledge Cells **530***d*, the system may insert (i.e. copy, store, etc.) the Knowledge Cell **800** from Knowledge Structuring Unit **520** into Collection of Knowledge Cells **530***d*, for example. On the other hand, if a substantially similar Knowledge Cell **800** is found in Collection of Knowledge Cells **530***d*, the system may optionally omit inserting the Knowledge Cell **800** from Knowledge Structuring Unit **520** as inserting a substantially similar Knowledge Cell **800** may not add much or any additional knowledge to the Collection of Knowledge Cells **530***d*, for example. Also, inserting a substantially similar Knowledge Cell **800** can optionally be omitted to save storage resources and limit the number of Knowledge Cells **800** that may later need to be processed or compared. Any features, functionalities, and embodiments of Similarity Comparison **125**, importance index (later described), similarity index (later described), and/or other disclosed elements can be utilized to facilitate determination of substantial or other similarity and whether to insert a newly structured Knowledge Cell **800** into Collection of Knowledge Cells **530***d*.

For example, the system can perform Similarity Comparisons **125** (later described) of Knowledge Cell **800***ba* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. In the case that a substantially similar match is found between Knowledge Cell **800***ba* and any of the Knowledge Cells **800** in Collection of Knowledge Cells **530***d*, the system may perform no action. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bb* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. In the case that a substantially similar match is not found, the system may insert a new Knowledge Cell **800** into Collection of Knowledge Cells **530***d* and copy Knowledge Cell **800***bb* into the inserted new Knowledge Cell **800**. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bc* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. In the case that a substantially similar match is found between Knowledge Cell **800***bc* and any of the Knowledge Cells **800** in Collection of Knowledge Cells **530***d*, the system may perform no action. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bd* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. In the case that a substantially similar match is not found, the system may insert a new Knowledge Cell **800** into Collection of Knowledge Cells **530***d* and copy Knowledge Cell **800***bd* into the inserted new Knowledge Cell **800**. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***be* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. In the case that a substantially similar match is not found, the system may insert a new Knowledge Cell **800** into Collection of Knowledge Cells **530***d* and copy Knowledge Cell **800***be* into the inserted new Knowledge Cell **800**. Applying any additional Knowledge Cells **800** from Knowledge Structuring Unit **520** onto Collection of Knowledge Cells **530***d* follows similar logic or process as the above-described.

Referring to FIG. **20**, an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Neural Network **530***a* is illustrated. Neural Network **530***a* includes a number of neurons or Nodes **852** interconnected by Connections **853** as previously described. Knowledge Cells **800** are shown instead of Nodes **852** to simplify the illustration as Node **852** includes a Knowledge Cell **800**, for example. Therefore, Knowledge Cells **800** and Nodes **852** can be used interchangeably herein depending on context. It should be noted that Node **852** may include other elements and/or functionalities instead of or in addition to Knowledge Cell **800**. In some aspects, Knowledge Cells **800** may be stored into or applied onto Neural Network **530***a* individually or collectively in a learning or training process. In some designs, Neural Network **530***a* comprises a number of Layers **854** each of which may include one or more Knowledge Cells **800**. Knowledge Cells **800** in successive Layers **854** can be connected by Connections **853**. Connection **853** may include or be associated with occurrence count, weight, and/or other parameter or data as previously described. Neural Network **530***a* may include any number of Layers **854** comprising any number of Knowledge Cells **800**. In some aspects, Neural Network **530***a* may store Knowledge Cells **800** interconnected by Connections **853** where following a path through the Neural Network **530***a* can later be used to enable autonomous Device **98** operation. It should be understood that, in some embodiments, Knowledge Cells **800** in one Layer **854** of Neural Network **530***a* need not be connected only with Knowledge Cells **800** in a successive Layer **854**, but also in any other Layer **854**, thereby creating shortcuts (i.e. shortcut Connections **853**, etc.) through Neural Network **530***a*. A Knowledge Cell **800** can also be connected to itself such as, for example, in recurrent neural networks. In general, any Knowledge Cell **800** can be connected with any other Knowledge Cell **800** anywhere else in Neural Network **530***a*. In further embodiments, back-propagation of any data or information can be implemented. In one example, back-propagation of similarity (i.e. similarity index, etc.) of compared Knowledge Cells **800** in a path through Neural Network **530***a* can be implemented. In another example, back-propagation of errors can be implemented. Such back-propagations can then be used to adjust occurrence counts and/or weights of Connections **853** for better future predictions, for example. Any other back-propagation can be

implemented for other purposes. Any combination of Nodes **852** (i.e. Nodes **852** comprising Knowledge Cells **800**, etc.), Connections **853**, Layers **854**, and/or other elements or techniques can be implemented in alternate embodiments. Neural Network **530***a* may include any type or form of a neural network known in art such as a feed-forward neural network, a back-propagating neural network, a recurrent neural network, a convolutional neural network, deep neural network, and/or others including a custom neural network.

In some embodiments, Knowledge Structuring Unit **520** structures or generates Knowledge Cells **800** and the system applies them onto Neural Network **530***a*, thereby implementing learning Device's **98** operation in circumstances including objects with various properties. The system can perform Similarity Comparisons **125** (later described) of a Knowledge Cell **800** from Knowledge Structuring Unit **520** with Knowledge Cells **800** in a Layer **854** of Neural Network **530***a*. If a substantially similar Knowledge Cell **800** is not found in the Layer **854** of Neural Network **530***a*, the system may insert (i.e. copy, store, etc.) the Knowledge Cell **800** from Knowledge Structuring Unit **520** into the Layer **854** of Neural Network **530***a*, and create a Connection **853** to the inserted Knowledge Cell **800** from a Knowledge Cell **800** in a prior Layer **854** including assigning an occurrence count to the new Connection **853**, calculating a weight of the new Connection **853**, and updating any other Connections **853** originating from the Knowledge Cell **800** in the prior Layer **854**. On the other hand, if a substantially similar Knowledge Cell **800** is found in the Layer **854** of Neural Network **530***a*, the system may update occurrence count and weight of Connection **853** to that Knowledge Cell **800** from a Knowledge Cell **800** in a prior Layer **854**, and update any other Connections **853** originating from the Knowledge Cell **800** in the prior Layer **854**.

For example, the system can perform Similarity Comparisons **125** (later described) of Knowledge Cell **800***ba* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Layer **854***a* of Neural Network **530***a*. In the case that a substantially similar match is found between Knowledge Cell **800***ba* and Knowledge Cell **800***ea*, the system may perform no action since Knowledge Cell **800***ea* is the initial Knowledge Cell **800**. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bb* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Layer **854***b* of Neural Network **530***a*. In the case that a substantially similar match is found between Knowledge Cell **800***bb* and Knowledge Cell **800***eb*, the system may update occurrence count and weight of Connection **853***e***1** between Knowledge Cell **800***ea* and Knowledge Cell **800***eb*, and update weights of other Connections **853** originating from Knowledge Cell **800***ea* as previously described. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bc* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Layer **854***c* of Neural Network **530***a*. In the case that a substantially similar match is not found, the system may insert Knowledge Cell **800***ec* into Layer **854***c* and copy Knowledge Cell **800***bc* into the inserted Knowledge Cell **800***ec*. The system may also create Connection **853***e***2** between Knowledge Cell **800***eb* and Knowledge Cell **800***ec* with occurrence count of 1 and weight calculated based on the occurrence count as previously described. The system may also update weights of other Connections **853** (one in this example) originating from Knowledge Cell **800***eb* as previously described. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bd* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Layer **854***d* of Neural Network **530***a*. In the case that a substantially similar match is not found, the system may insert Knowledge Cell **800***ed* into Layer **854***d* and copy Knowledge Cell **800***bd* into the inserted Knowledge Cell **800***ed*. The system may also create Connection **853***e***3** between Knowledge Cell **800***ec* and Knowledge Cell **800***ed* with occurrence count of 1 and weight of 1. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***be* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Layer **854***e* of Neural Network **530***a*. In the case that a substantially similar match is not found, the system may insert Knowledge Cell **800***ee* into Layer **854***e* and copy Knowledge Cell **800***be* into the inserted Knowledge Cell **800***ee*. The system may also create Connection **853***e***4** between Knowledge Cell **800***ed* and Knowledge Cell **800***ee* with occurrence count of 1 and weight of 1. Applying any additional Knowledge Cells **800** from Knowledge Structuring Unit **520** onto Neural Network **530***a* follows similar logic or process as the above-described.

Referring now to Similarity Comparison **125**, Similarity Comparison **125** comprises the functionality for comparing or matching Knowledge Cells **800** or portions thereof, and/or other functionalities. Similarity Comparison **125** comprises the functionality for comparing or matching Collections of Object Representations **525** or portions thereof. Similarity Comparison **125** comprises the functionality for comparing or matching streams of Collections of Object Representations **525** or portions thereof. Similarity Comparison **125** comprises the functionality for comparing or matching Object Representations **625** or portions thereof. Similarity Comparison **125** comprises the functionality for comparing or matching Object Properties **630** or portions thereof. Similarity Comparison **125** comprises the functionality for comparing or matching Instruction Sets **526**, Extra Info **527**, text (i.e. characters, words, phrases, etc.), numbers, and/or other elements or portions thereof. Similarity Comparison **125** may include functions, rules, and/or logic for performing matching or comparisons and for determining that while a perfect match is not found, a partial or similar match has been found. In some aspects, a partial match may include a substantially or otherwise similar match, and vice versa. Therefore, these terms may be used interchangeably herein depending on context. As such, Similarity Comparison **125** may include determining substantial similarity or substantial match of compared elements. Although, substantial similarity or substantial match is frequently used herein, it should be understood that any level of similarity, however high or low, may be utilized as defined by the rules (i.e. thresholds, etc.) for similarity. The rules for similarity or similar match can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. In some designs, Similarity Comparison **125** comprises the functionality to automatically define appropriately strict rules for determining similarity of the compared elements. Similarity Comparison **125** can therefore set, reset, and/or adjust the strictness of the rules for finding or determining similarity of the compared elements, thereby fine tuning Similarity Comparison **125** so that the rules for determining similarity are appropriately strict. In some aspects, the rules for determining similarity may include a similarity threshold. As such, Similarity Comparison **125** can determine similarity of compared elements if their similarity exceeds a similarity threshold. In other aspects, the rules for determining similarity may include a difference threshold. As such, Similarity Comparison **125** can determine similarity of compared elements if

their difference is lower than a difference threshold. In further aspects, the rules for determining similarity may include other thresholds. Similarity Comparison **125** enables comparing circumstances including objects with various properties and determining their similarity or match. In one example, a circumstance including an object detected at a distance of 8 m and an angle/bearing of 64° relative to Device **98** may be found similar or matching by Similarity Comparison **125** to a circumstance including the same or similar object detected at a distance of 8.6 m and an angle/bearing of 59° relative to Device **98**. In another example, a circumstance including an object detected as a passenger vehicle may be found similar or matching by Similarity Comparison **125** to a circumstance including an object detected as a sport utility vehicle. In general, any one or more properties (i.e. existence, type, identity, distance, bearing/angle, location, shape/size, activity, etc.) of one or more objects can be utilized for determining similarity or match of circumstances including objects with various properties. Therefore, Similarity Comparison **125** provides flexibility in comparing and determining similarity of a variety of possible circumstances of Device **98**.

In some embodiments where compared Knowledge Cells **800** include a single Collection of Object Representations **525**, in determining similarity of Knowledge Cells **800**, Similarity Comparison **125** can perform comparison of individual Collections of Object Representations **525** or portions (i.e. Object Representations **625**, Object Properties **630**, etc.) thereof such as comparison of Collection of Object Representations **525** or portions thereof from one Knowledge Cell **800** with Collection of Object Representations **525** or portions thereof from another Knowledge Cell **800**. In some aspects, total equivalence is achieved when Collection of Object Representations **525** or portions thereof from one Knowledge Cell **800** matches Collection of Object Representations **525** or portions thereof from another Knowledge Cell **800**. If total equivalence is not found, Similarity Comparison **125** may attempt to determine substantial or other similarity of compared Knowledge Cells **800**.

In some embodiments, in determining substantial similarity of individually compared Collections of Object Representations **525** (i.e. Collections of Object Representations **525** from the compared Knowledge Cells **800**, etc.), Similarity Comparison **125** can compare one or more Object Representations **625** or portions (i.e. Object Properties **630**, etc.) thereof from one Collection of Object Representations **525** with one or more Object Representations **625** or portions thereof from another Collection of Object Representations **525**. In some aspects, total equivalence is found when all Object Representations **625** or portions thereof from one Collection of Object Representations **525** match all Object Representations **625** or portions thereof from another Collection of Object Representations **525**. In other aspects, if total equivalence is not found, Similarity Comparison **125** may attempt to determine substantial similarity of compared Collections of Object Representations **525**. In one example, substantial similarity can be achieved when most of the Object Representations **625** or portions thereof from the compared Collections of Object Representations **525** match or substantially match. In another example, substantial similarity can be achieved when at least a threshold number (i.e. 1, 2, 4, 7, 18, etc.) or percentage (i.e. 41%, 62%, 79%, 85%, 93%, etc.) of Object Representations **625** or portions thereof from the compared Collections of Object Representations **525** match or substantially match. Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching Object Representations **625** or portions thereof from the compared Collections of Object Representations **525** exceeds a threshold number (i.e. 1, 2, 4, 7, 18, etc.) or a threshold percentage (i.e. 41%, 62%, 79%, 85%, 93%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or percentage of Object Representations **625** or portions thereof from the compared Collections of Object Representations **525** match or substantially match. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In some aspects, Similarity Comparison **125** can utilize the importance (i.e. as indicated by importance index [later described], etc.) of Object Representations **625** or portions thereof for determining substantial similarity of Collections of Object Representations **525**. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to more important Object Representations **625** or portions thereof such as Object Representations **625** representing near Objects **615**, Object Representations **625** representing large Objects **615**, etc., thereby tolerating mismatches in less important Object Representations **625** or portions thereof such as Object Representations **625** representing distant Objects **615**, Object Representations **625** representing small Objects **615**, etc. In general, any Object Representation **625** or portion thereof can be assigned higher or lower importance. In further aspects, Similarity Comparison **125** can omit some of the Object Representations **625** or portions thereof from the comparison in determining substantial similarity of Collections of Object Representations **525**. In one example, Object Representations **625** representing distant Objects **615** can be omitted from comparison. In another example, Object Representations **625** representing small Objects **615** can be omitted from comparison. In general, any Object Representation **625** or portion thereof can be omitted from comparison depending on implementation.

Similarity Comparison **125** can automatically adjust (i.e. increase or decrease) the strictness of the rules for determining substantial similarity of Collections of Object Representations **525**. In some aspects, such adjustment in strictness can be done by Similarity Comparison **125** in response to determining that total equivalence of compared Collections of Object Representations **525** had not been found. Similarity Comparison **125** can keep adjusting the strictness rules until a substantial similarity is found. All the rules or settings of substantial similarity can be set, reset, or adjusted by Similarity Comparison **125** in response to another strictness level determination. For example, Similarity Comparison **125** may attempt to find a match or substantial match in a certain percentage (i.e. 81%, etc.) of Object Representations **625** or portions thereof from the compared Collections of Object Representations **525**. If the comparison does not determine substantial similarity of compared Collections of Object Representations **525**, Similarity Comparison **125** may decide to decrease the strictness of the rules. In response, Similarity Comparison **125** may attempt to find fewer matching or substantially matching Object Representations **625** or portions thereof than in the previous attempt using stricter rules. If the comparison still does not determine substantial similarity of compared Collections of Object Representations **525**, Similarity Comparison **125** may further decrease the strictness (i.e. down to a certain minimum strictness or threshold, etc.) by requiring fewer Object Representations **625** or portions thereof to match or

substantially match, thereby further increasing a chance of finding substantial similarity in compared Collections of Object Representations **525**.

Where a reference to Object Representation **625** is used herein it should be understood that a portion of Object Representation **625** (i.e. Object Property **630**, etc.) or a plurality of Object Representations **625** can be used instead of or in addition to the Object Representation **625**. In one example, instead of or in addition to Object Representation **625**, Object Properties **630** and/or other portions that constitute an Object Representation **625** can be compared. In another example, instead of or in addition to Object Representation **625**, plurality of Object Representations **625** can be compared. As such, any operations, rules, logic, and/or functions operating on Object Representation **625** may similarly apply to any portion of Object Representation **625** and/or a plurality of Object Representations **625** as applicable. In general, whole Object Representations **625**, portions of Object Representations **625**, and/or pluralities of Object Representations **625**, including any operations thereon, can be combined to arrive at desired results. Some or all of the above-described rules, logic, and/or techniques can be utilized alone or in combination with each other or with other rules, logic, and/or techniques. One of ordinary skill in art will recognize that other techniques known in art for determining similarity of Object Representations **625** and/or other data that would be too voluminous to describe are within the scope of this disclosure.

In some embodiments, in determining substantial similarity of Object Representations **625** (i.e. Object Representations **625** from the compared Collections of Object Representations **525**, etc.), Similarity Comparison **125** can compare Object Properties **630** or portions (i.e. characters, words, numbers, etc.) thereof from one Object Representation **625** with Object Properties **630** or portions thereof from another Object Representation **625**. In some aspects, total equivalence is found when all Object Properties **630** or portions thereof of one Object Representation **625** match all Object Properties **630** or portions thereof of another Object Representation **625**. In other aspects, if total equivalence is not found, Similarity Comparison **125** may attempt to determine substantial similarity of compared Object Representations **625**. In one example, substantial similarity can be achieved when most of the Object Properties **630** or portions thereof from the compared Object Representations **625** match or substantially match. In another example, substantial similarity can be achieved when at least a threshold number (i.e. 1, 2, 3, 6, 11, etc.) or percentage (i.e. 55%, 61%, 78%, 82%, 99%, etc.) of Object Properties **630** or portions thereof from the compared Object Representations **625** match or substantially match. Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching Object Properties **630** or portions thereof from the compared Object Representations **625** exceeds a threshold number (i.e. 1, 2, 3, 6, 11, etc.) or a threshold percentage (i.e. 55%, 61%, 78%, 82%, 99%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or percentage of Object Properties **630** or portions thereof from the compared Object Representations **625** match or substantially match. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In further aspects, Similarity Comparison **125** can utilize Categories 635 associated with Object Properties **630** for determining substantial similarity of Object Representations **625**. In one example, Object Properties **630** or portions thereof from the compared Object Representations **625** in a same Category **635** may be compared. This way, Object Properties **630** or portions thereof can be compared with their own peers. In one instance, Object Properties **630** or portions thereof from the compared Object Representations **625** in Category **635** "Type" may be compared. Any text comparison technique can be utilized in such comparing. In another instance, Object Properties **630** or portions thereof from the compared Object Representations **625** in Category **635** "Distance" or "Bearing" may be compared. Any number comparison technique can be utilized in such comparing. In a further instance, Object Properties **630** or portions thereof from the compared Object Representations **625** in Category **635** "Shape" may be compared. Any model, point cloud, or other computer construct comparison technique can be utilized in such comparing. In further aspects, Similarity Comparison **125** can utilize the importance (i.e. as indicated by importance index [later described], etc.) of Object Properties **630** or portions thereof for determining substantial similarity of Object Representations **625**. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to more important Object Properties **630** or portions thereof such as Object Properties **630** or portions thereof in Categories 635 "Type", "Distance", "Bearing", etc., thereby tolerating mismatches in less important Object Properties **630** or portions thereof such as Object Properties **630** or portions thereof in Categories 635 "Identity", "Shape", etc. In general, any Object Property **630** or portion thereof can be assigned higher or lower importance. In further aspects, Similarity Comparison **125** can omit some of the Object Properties **630** or portions thereof from the comparison in determining substantial similarity of Object Representations **625**. In one example, Object Properties **630** or portions thereof in Category **635** "Identity" can be omitted from comparison. In another example, Object Properties **630** or portions thereof in Category **635** "Shape" can be omitted from comparison. In general, any Object Property **630** or portion thereof can be omitted from comparison depending on implementation.

Similarity Comparison **125** can automatically adjust (i.e. increase or decrease) the strictness of the rules for determining substantial similarity of Object Representations **625**. In some aspects, such adjustment in strictness can be done by Similarity Comparison **125** in response to determining that total equivalence of compared Object Representations **625** had not been found. Similarity Comparison **125** can keep adjusting the strictness rules until a substantial similarity is found. All the rules or settings of substantial similarity can be set, reset, or adjusted by Similarity Comparison **125** in response to another strictness level determination. For example, Similarity Comparison **125** may attempt to find a match or substantial match in a certain percentage (i.e. 87%, etc.) of Object Properties **630** or portions thereof from the compared Object Representations **625**. If the comparison does not determine substantial similarity of compared Object Representations **625**, Similarity Comparison **125** may decide to decrease the strictness of the rules. In response, Similarity Comparison **125** may attempt to find fewer matching or substantially matching Object Properties **630** or portions thereof than in the previous attempt using stricter rules. If the comparison still does not determine substantial similarity of compared Object Representations **625**, Similarity Comparison **125** may further decrease the strictness (i.e. down to a certain minimum strictness or threshold, etc.) by requiring fewer Object Properties **630** or portions thereof to match or substantially

match, thereby further increasing a chance of finding substantial similarity in compared Object Representations **625**. In further aspects, an adjustment in strictness can be done by Similarity Comparison **125** in response to determining that multiple substantially similar Object Representations **625** had been found. Similarity Comparison **125** can keep adjusting the strictness of the rules until a best of the substantially similar Object Representations **625** is found. For example, Similarity Comparison **125** may attempt to find a match or substantial match in a certain percentage (i.e. 65%, etc.) of Object Properties **630** or portions thereof from the compared Object Representations **625**. If the comparison determines a number of substantially similar Object Representations **625**, Similarity Comparison **125** may decide to increase the strictness of the rules to decrease the number of substantially similar Object Representations **625**. In response, Similarity Comparison **125** may attempt to find more matching or substantially matching Object Properties **630** or portions thereof in addition to the earlier found Object Properties **630** or portions thereof to limit the number of substantially similar Object Representations **625**. If the comparison still provides more than one substantially similar Object Representation **625**, Similarity Comparison **125** may further increase the strictness by requiring additional Object Properties **630** or portions thereof to match or substantially match, thereby further narrowing the number of substantially similar Object Representations **625** until a best substantially similar Object Representation **625** is found.

Where a reference to Object Property **630** is used herein it should be understood that a portion of Object Property **630** or a plurality of Object Properties **630** can be used instead of or in addition to the Object Property **630**. In one example, instead of or in addition to Object Property **630**, characters, words, numbers, and/or other portions that constitute an Object Property **630** can be compared. In another example, instead of or in addition to Object Property **630**, a plurality of Object Properties **630** can be compared. As such, any operations, rules, logic, and/or functions operating on Object Property **630** may similarly apply to any portion of Object Property **630** and/or a plurality of Object Properties **630** as applicable. In general, whole Object Properties **630**, portions of Object Properties **630**, and/or pluralities of Object Properties **630**, including any operations thereon, can be combined to arrive at desired results. Some or all of the above-described rules, logic, and/or techniques can be utilized alone or in combination with each other or with other rules, logic, and/or techniques. One of ordinary skill in art will recognize that other techniques known in art for determining similarity of Object Properties **630** and/or other data that would be too voluminous to describe are within the scope of this disclosure.

In some embodiments where compared Knowledge Cells **800** include a stream of Collections of Object Representations **525**, in determining similarity of Knowledge Cells **800**, Similarity Comparison **125** can perform collective comparison of Collections of Object Representations **525** or portions (i.e. Object Representations **625**, Object Properties **630**, etc.) thereof such as comparison of a stream of Collections of Object Representations **525** or portions thereof from one Knowledge Cell **800** with a stream of Collections of Object Representations **525** or portions thereof from another Knowledge Cell **800**. Similarity Comparison **125** of collectively compared Collections of Object Representations **525** or portions thereof may include any features, functionalities, and embodiments of the previously described Similarity Comparison **125** of individually compared Collections of Object Representations **525** or portions thereof. In some aspects, total equivalence is found when all Collections of Object Representations **525** or portions thereof from one Knowledge Cell **800** match all Collections of Object Representations **525** or portions thereof from another Knowledge Cell **800**. If total equivalence is not found, Similarity Comparison **125** may attempt to determine substantial or other similarity of compared Knowledge Cells **800**. In one example, substantial similarity can be achieved when most of the Collections of Object Representations **525** or portions thereof from the compared Knowledge Cells **800** match or substantially match. In another example, substantial similarity can be achieved when at least a threshold number (i.e. 1, 2, 4, 9, 33, 138, etc.) or percentage (i.e. 39%, 58%, 77%, 88%, 94%, etc.) of Collections of Object Representations **525** or portions thereof from the compared Knowledge Cells **800** match or substantially match. Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching Collections of Object Representations **525** or portions thereof from the compared Knowledge Cells **800** exceeds a threshold number (i.e. 1, 2, 4, 9, 33, 138, etc.) or a threshold percentage (i.e. 39%, 58%, 77%, 88%, 94%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or percentage of Collections of Object Representations **525** or portions thereof from the compared Knowledge Cells **800** match or substantially match. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In some aspects, Similarity Comparison **125** can utilize the importance (i.e. as indicated by importance index [later described], etc.) of Collections of Object Representations **525** or portions thereof for determining substantial similarity of Knowledge Cells **800**. In one example, substantial similarity can be achieved when matches or substantial matches are found with respect to more important Collections of Object Representations **525** or portions thereof such as more substantive or larger Collections of Object Representations **525** (i.e. Collections of Object Representations **525** comprising a higher number of Object Representations **625**, etc.) or portions thereof, etc., thereby tolerating mismatches in less important Collections of Object Representations **525** or portions thereof such as less substantive or smaller Collections of Object Representations **525** (i.e. Collections of Object Representations **525** comprising a lower number of Object Representations **625**, etc.) or portions thereof, etc. In general, any Collection of Object Representations **525** or portion thereof can be assigned higher or lower importance. In other aspects, Similarity Comparison **125** can utilize the order of Collections of Object Representations **525** or portions thereof for determining substantial similarity of Knowledge Cells **800**. In one example, substantial similarity can be achieved when matches or substantial matches are found in earlier Collections of Object Representations **525** or portions thereof from the compared Knowledge Cells **800**, thereby tolerating mismatches in later Collections of Object Representations **525** or portions thereof. In another example, substantial similarity can be achieved when matches or substantial matches are found in corresponding (i.e. similarly ordered, temporally related, etc.) Collections of Object Representations **525** or portions thereof from the compared Knowledge Cells **800**. In one instance, a 94^{th} Collection of Object Representations **525** or portions thereof from one Knowledge Cell **800** can be compared with a 94^{th} Collection of Object Representations **525** or portions thereof from another Knowledge Cell **800**. In another instance, a 94^{th} Collection

of Object Representations **525** or portions thereof from one Knowledge Cell **800** can be compared with a number of Collections of Object Representations **525** or portions thereof around (i.e. preceding and/or following) a 94^{th} Collection of Object Representations **525** from another Knowledge Cell **800**. This way, flexibility can be implemented in finding a substantially similar Collection of Object Representations **525** or portions thereof if the Collections of Object Representations **525** or portions thereof in the compared Knowledge Cells **800** are not perfectly aligned. In a further instance, Similarity Comparison **125** can utilize Dynamic Time Warping (DTW) and/or other techniques known in art for comparing and/or aligning temporal sequences (i.e. streams of Collections of Object Representations **525** or portions thereof, etc.) that may vary in time or speed. In further aspects, Similarity Comparison **125** can omit some of the Collections of Object Representations **525** or portions thereof from the comparison in determining substantial similarity of Knowledge Cells **800**. In one example, less substantive or smaller Collections of Object Representations **525** or portions thereof can be omitted from comparison. In another example, some or all Collections of Object Representations **525** or portions thereof related to a specific time period can be omitted from comparison. In general, any Collection of Object Representations **525** or portion thereof can be omitted from comparison depending on implementation.

Similarity Comparison **125** can automatically adjust (i.e. increase or decrease) the strictness of the rules for determining substantial similarity of Knowledge Cells **800**. In some aspects, such adjustment in strictness can be done by Similarity Comparison **125** in response to determining that total equivalence of compared Knowledge Cells **800** had not been found. Similarity Comparison **125** can keep adjusting the strictness of the rules until substantial similarity is found. All the rules or settings of substantial similarity can be set, reset, or adjusted by Similarity Comparison **125** in response to another strictness level determination. For example, Similarity Comparison **125** may attempt to find a match or substantial match in a certain percentage (i.e. 92%, etc.) of Collections of Object Representations **525** or portions thereof from the compared Knowledge Cells **800**. If the comparison does not determine substantial similarity of compared Knowledge Cells **800**, Similarity Comparison **125** may decide to decrease the strictness of the rules. In response, Similarity Comparison **125** may attempt to find fewer matching or substantially matching Collections of Object Representations **525** or portions thereof than in the previous attempt using stricter rules. If the comparison still does not determine substantial similarity of compared Knowledge Cells **800**, Similarity Comparison **125** may further decrease (i.e. down to a certain minimum strictness or threshold, etc.) the strictness by requiring fewer Collections of Object Representations **525** or portions thereof to match or substantially match, thereby further increasing a chance of finding substantial similarity in compared Knowledge Cells **800**. In further aspects, an adjustment in strictness can be done by Similarity Comparison **125** in response to determining that multiple substantially similar Knowledge Cells **800** had been found. Similarity Comparison **125** can keep adjusting the strictness of the rules until a best of the substantially similar Knowledge Cells **800** is found. For example, Similarity Comparison **125** may attempt to find a match or substantial match in a certain percentage (i.e. 71%, etc.) of Collections of Object Representations **525** or portions thereof from the compared Knowledge Cells **800**. If the comparison determines a number of substantially similar Knowledge Cells **800**, Similarity Comparison **125** may decide to increase the strictness of the rules to decrease the number of substantially similar Knowledge Cells **800**. In response, Similarity Comparison **125** may attempt to find more matching or substantially matching Collections of Object Representations **525** or portions thereof in addition to the earlier found Collections of Object Representations **525** or portions thereof to limit the number of substantially similar Knowledge Cells **800**. If the comparison still provides more than one substantially similar Knowledge Cell **800**, Similarity Comparison **125** may further increase the strictness by requiring additional Collections of Object Representations **525** or portions thereof to match or substantially match, thereby further narrowing the number of substantially similar Knowledge Cells **800** until a best substantially similar Knowledge Cell **800** is found.

Some or all of the aforementioned rules, logic, and/or techniques for determining substantial similarity of Knowledge Cells **800** can be utilized alone or in combination with each other or with other rules, logic, and/or techniques. One of ordinary skill in art will recognize that other techniques known in art for determining similarity of Knowledge Cells **800** and/or other data that would be too voluminous to describe are within the scope of this disclosure.

In any of the comparisons involving numbers such as, for example, Object Properties **630** including numbers (i.e. distances, bearings/angles, etc.), Similarity Comparison **125** can compare a number from one Object Property **630** with a number from another Object Property **630**. In some aspects, total equivalence is found when the number from one Object Property **630** equals the number from another Object Property **630**. In other aspects, if total equality is not found, Similarity Comparison **125** may attempt to determine substantial similarity of the compared numbers using a tolerance or threshold for determining a match. In some aspects, Similarity Comparison **125** can utilize a threshold for acceptable number difference in determining a match of compared numbers. For example, a threshold for acceptable number difference (i.e. absolute difference, etc.) can be set at 10. Therefore, **130** matches or is sufficiently similar to **135** because the number difference (i.e. 5 in this example) is lower than the threshold for acceptable number difference (i.e. 10 in this example, etc.). Furthermore, **130** does not match or is not sufficiently similar to **143** because the number difference (i.e. 13 in this example) is greater than the threshold for acceptable number difference. Any other threshold for acceptable number difference can be used such as 0.024, 1, 8, 15, 77, 197, 2438, 728322, and/or others. In other aspects, Similarity Comparison **125** can utilize a threshold for acceptable percentage difference in determining a match of compared numbers. For example, a threshold for acceptable percentage difference can be set at 10%. Therefore, **100** matches or is sufficiently similar to **106** because the percentage difference (i.e. 6% in this example) is lower than the threshold for acceptable percentage difference (i.e. 10% in this example). Furthermore, **100** does not match or is not sufficiently similar to **84** because the percentage difference (i.e. 16% in this example) is higher than the threshold for acceptable percentage difference. Any other threshold for acceptable percentage difference can be used such as 0.68%, 1%, 3%, 11%, 33%, 69%, 87%, and/or others. The aforementioned thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. Other techniques known in art for comparing numbers can be utilized herein.

In any of the comparisons involving text such as, for example, Object Properties **630** including text (i.e. types, identities, etc.), Similarity Comparison **125** can compare words, characters, and/or other text from one Object Property **630** with words, characters, and/or other text from another Object Property **630**. In some aspects, total equivalence is found when all words, characters, and/or other text from one Object Property **630** match all words, characters, and/or other text from another Object Property **630**. In other aspects, if total equivalence is not found, Similarity Comparison **125** may attempt to determine substantial similarity of compared Object Properties **630**. In one example, substantial similarity can be achieved when most of the words, characters, and/or other text from the compared Object Properties **630** match or substantially match. In another example, substantial similarity can be achieved when at least a threshold number (i.e. 1, 2, 3, 4, 7, 11, etc.) or percentage (i.e. 38%, 63%, 77%, 84%, 98%, etc.) of words, characters, and/or other text from the compared Object Properties **630** match or substantially match. Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching words, characters, and/or other text from the compared Object Properties **630** exceeds a threshold number (i.e. 1, 2, 3, 4, 7, 11, etc.) or a threshold percentage (i.e. 48%, 63%, 77%, 84%, 98%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or percentage of words, characters, and/or other text from the compared Object Properties **630** match or substantially match. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In further aspects, Similarity Comparison **125** can utilize the importance (i.e. as indicated by importance index [later described], etc.) of words, characters, and/or other text for determining substantial similarity of Object Properties **630**. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to more important words, characters, and/or other text such as longer words and/or other text, thereby tolerating mismatches in less important words, characters, and/or other text such as shorter words and/or other text. In general, any word, character, and/or other text can be assigned higher or lower importance. In further aspects, Similarity Comparison **125** can utilize the order of words, characters, and/or other text for determining substantial similarity of Object Properties **630**. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to front-most words, characters, and/or other text, thereby tolerating mismatches in later words, characters, and/or other text. In further aspects, Similarity Comparison **125** can utilize semantic conversion to account for variations of words and/or other text. In one example, Object Property **630** may include a word "house". In addition to searching for the exact word in a compared Object Property **630**, Similarity Comparison **125** can employ semantic conversion and attempt to match "home", "residence", "dwelling", "place", or other semantically similar variations of the word with a meaning "house". In another example, Object Property **630** may include a word "buy". In addition to searching for the exact word in a compared Object Property **630**, Similarity Comparison **125** can employ semantic conversion and attempt to match "buying", "bought", or other semantically similar variations of the word with a meaning "buy" in different tenses. Any other grammatical analysis or transformation can be utilized to cover the full scope of word and/or other text variations. In some designs, semantic conversion can be implemented using a thesaurus or dictionary. In another example, semantic conversion can be implemented using a table where each row comprises semantically similar variations of a word and/or other text. In further aspects, Similarity Comparison **125** can utilize a language model for understanding or interpreting the concepts contained in the words and/or other text and compare the concepts instead of or in addition to the words and/or other text. Examples of language models include unigram model, n-gram model, neural network language model, bag of words model, and/or others. Any of the techniques for matching of words can similarly be used for matching of concepts. In further aspects, Similarity Comparison **125** can omit some of the words, characters, and/or other text from the comparison in determining substantial similarity of Object Properties **630**. In one example, rear-most words, characters, and/or other text can be omitted from comparison. In another example, shorter words and/or other text can be omitted from comparison. In general, any word, character, and/or other text can be omitted from comparison depending on implementation. Other techniques known in art for comparing words, characters, and/or other text can be utilized herein.

In some embodiments, Similarity Comparison **125** can compare one or more Extra Info **527** (i.e. time information, location information, computed information, contextual information, and/or other information, etc.) in addition to or instead of comparing Collections of Object Representations **525** or portions thereof in determining substantial similarity of Knowledge Cells **800**. Extra Info **527** can be set to be less, equally, or more important (i.e. as indicated by importance index [later described], etc.) than Collections of Object Representations **525**, Object Representations **625**, Object Properties **630**, and/or other elements in the comparison. Since Extra Info **527** may include any contextual or other information that can be useful in determining similarity of any compared elements, Extra Info **527** can be used to enhance any of the aforementioned similarity determinations as applicable.

In some embodiments, Similarity Comparison **125** can also compare one or more Instruction Sets **526** in addition to or instead of comparing Collections of Object Representations **525** or portions thereof in determining substantial similarity of Knowledge Cells **800**. In some aspects, Similarity Comparison **125** can compare portions of Instruction Sets **526** to determine substantial or other similarity of Instruction Sets **526**. Similar to the above-described thresholds, thresholds for the number or percentage of matching portions of the compared Instruction Sets **526** can be utilized in determining substantial or other similarity of the compared Instruction Sets **526**. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In other aspects, Similarity Comparison **125** can compare text (i.e. characters, words, phrases, etc.), numbers, or other data (i.e. bits, etc.) to determine substantial or other similarity of Instruction Sets **526**. Any other comparison technique can be utilized in comparing Instruction Sets **526** in alternate implementations. Instruction Sets **526** can be set to be less, equally, or more important (i.e. as indicated by importance index [later described], etc.) than Collections of Object Representations **525**, Object Representations **625**, Object Properties **630**, Extra Info **527**, and/or other elements in the comparison.

In some embodiments, an importance index (not shown) or other importance ranking technique can be used in any of

the previously described comparisons or other processing involving elements of different importance. Importance index indicates importance of the element to or with which the index is assigned or associated. For example, importance index may indicate importance of a Knowledge Cell **800**, Collection of Object Representations **525**, Object Representation **625**, Object Property **630**, Instruction Set **526**, Extra Info **527**, and/or other element to or with which the index is assigned or associated. In some aspects, importance index on a scale from 0 to 1 can be utilized, although, any other range can also be utilized. Importance index can be stored in or associated with the element to which the index pertains. Importance indexes of various elements can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. In one example, a higher Importance index can be assigned to more substantive or larger Collections of Object Representations **525** (i.e. Collections of Object Representations **525** comprising a higher number of Object Representations **625**, etc.). In another example, a higher importance index can be assigned to Object Representations **625** representing closer, larger, and/or other Objects **615**. Any importance index can be assigned to or associated with any element described herein depending on implementation. Any importance ranking technique can be utilized as or instead of importance index in alternate embodiments.

In some embodiments, Similarity Comparison **125** may generate a similarity index (not shown) for any of the compared elements. Similarity index indicates how well an element is matched with another element. For example, similarity index indicates how well a Knowledge Cell **800**, Collection of Object Representations **525**, Object Representation **625**, Object Property **630**, Instruction Set **526**, Extra Info **527**, and/or other element is matched with a compared element. In some aspects, similarity index on a scale from 0 to 1 can be utilized, although, any other range can also be utilized. Similarity index can be generated by Similarity Comparison **125** whether substantial or other similarity between the compared elements is achieved or not. In one example, similarity index can be determined for a Knowledge Cell **800** based on a ratio/percentage of matched or substantially matched Collections of Object Representations **525** relative to the number of Collections of Object Representations **525** in the compared Knowledge Cell **800**. Specifically, similarity index of 0.91 is determined if 91% of Collections of Object Representations **525** of one Knowledge Cell **800** match or substantially match Collections of Object Representations **525** of another Knowledge Cell **800**. In some designs, importance (i.e. as indicated by importance index, etc.) of one or more Collections of Object Representations **525** can be included in the calculation of a weighted similarity index. Similar determination of similarity index can be implemented with Collections of Object Representations **525**, Object Representations **625**, Object Properties **630**, Instruction Sets **526**, Extra Info **527**, and/or other elements or portions thereof. Any combination of the aforementioned similarity index determinations or calculations can be utilized in alternate embodiments. Any similarity ranking technique can be utilized to determine or calculate similarity index in alternate embodiments.

Referring to FIG. **21**, an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Neural Network **530***a* comprising shortcut Connections **853** is illustrated. In some designs, Knowledge Cells **800** in one Layer **854** of Neural Network **530***a* can be connected with Knowledge Cells **800** in any Layer **854**, not only in a successive Layer **854**, thereby creating shortcuts (i.e. shortcut Connections **853**, etc.) through Neural Network **530***a*. In some aspects, creating a shortcut Connection **853** can be implemented by performing Similarity Comparisons **125** of a Knowledge Cell **800** from Knowledge Structuring Unit **520** with Knowledge Cells **800** in any Layer **854** when applying (i.e. storing, copying, etc.) the Knowledge Cell **800** from Knowledge Structuring Unit **520** onto Neural Network **530***a*. Once created, shortcut Connections **853** enable a wider variety of Knowledge Cells **800** to be considered when selecting a path through Neural Network **530***a*. In some embodiments, Knowledge Structuring Unit **520** structures or generates Knowledge Cells **800** and the system applies them onto Neural Network **530***a*, thereby implementing learning Device's **98** operation in circumstances including objects with various properties. The system can perform Similarity Comparisons **125** of a Knowledge Cell **800** from Knowledge Structuring Unit **520** with Knowledge Cells **800** in one or more Layers **854** of Neural Network **530***a*. If a substantially similar Knowledge Cell **800** is not found in the one or more Layers **854** of Neural Network **530***a*, the system may insert (i.e. copy, store, etc.) the Knowledge Cell **800** from Knowledge Structuring Unit **520** into a Layer **854** of Neural Network **530***a*, and create a Connection **853** to the inserted Knowledge Cell **800** from a prior Knowledge Cell **800** including assigning an occurrence count to the new Connection **853**, calculating a weight of the new Connection **853**, and updating any other Connections **853** originating from the prior Knowledge Cell **800**. On the other hand, if a substantially similar Knowledge Cell **800** is found in the one or more Layers **854** of Neural Network **530***a*, the system may update occurrence count and weight of Connection **853** to that Knowledge Cell **800** from a prior Knowledge Cell **800**, and update any other Connections **853** originating from the prior Knowledge Cell **800**. Any of the previously described and/or other techniques for comparing, inserting, updating, and/or other operations on Knowledge Cells **800**, Connections **853**, Layers **854**, and/or other elements can similarly be utilized in Neural Network **530***a* that comprises shortcut Connections **853**.

Referring to FIG. **22**, an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Graph **530***b* is illustrated. In some aspects, any Knowledge Cell **800** can be connected with any other Knowledge Cell **800** in Graph **530***b*. In other aspects, any Knowledge Cell **800** can be connected with itself and/or any other Knowledge Cell **800** in Graph **530***b*. In some embodiments, Knowledge Structuring Unit **520** structures or generates Knowledge Cells **800** and the system applies (i.e. store, copy, etc.) them onto Graph **530***b*, thereby implementing learning Device's **98** operation in circumstances including objects with various properties. The system can perform Similarity Comparisons **125** of a Knowledge Cell **800** from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Graph **530***b*. If a substantially similar Knowledge Cell **800** is not found in Graph **530***b*, the system may insert (i.e. copy, store, etc.) the Knowledge Cell **800** from Knowledge Structuring Unit **520** into Graph **530***b*, and create a Connection **853** to the inserted Knowledge Cell **800** from a prior Knowledge Cell **800** including assigning an occurrence count to the new Connection **853**, calculating a weight of the new Connection **853**, and updating any other Connections **853** originating from the prior Knowledge Cell **800**. On the other hand, if a

substantially similar Knowledge Cell **800** is found in Graph **530***b*, the system may update occurrence count and weight of Connection **853** to that Knowledge Cell **800** from a prior Knowledge Cell **800**, and update any other Connections **853** originating from the prior Knowledge Cell **800**. Any of the previously described and/or other techniques for comparing, inserting, updating, and/or other operations on Knowledge Cells **800**, Connections **853**, and/or other elements can similarly be utilized in Graph **530***b*.

For example, the system can perform Similarity Comparisons **125** of Knowledge Cell **800***ba* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Graph **530***b*. In the case that a substantially similar match is not found, the system may insert Knowledge Cell **800***ha* into Graph **530***b* and copy Knowledge Cell **800***ba* into the inserted Knowledge Cell **800***ha*. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bb* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Graph **530***b*. In the case that a substantially similar match is found between Knowledge Cell **800***bb* and Knowledge Cell **800***hb*, the system may create Connection **853***h***1** between Knowledge Cell **800***ha* and Knowledge Cell **800***hb* with occurrence count of 1 and weight of 1. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bc* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Graph **530***b*. In the case that a substantially similar match is found between Knowledge Cell **800***bc* and Knowledge Cell **800***hc*, the system may update occurrence count and weight of Connection **853***h***2** between Knowledge Cell **800***hb* and Knowledge Cell **800***hc*, and update weights of other outgoing Connections **853** (one in this example) originating from Knowledge Cell **800***hb* as previously described. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bd* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Graph **530***b*. In the case that a substantially similar match is not found, the system may insert Knowledge Cell **800***hd* into Graph **530***b* and copy Knowledge Cell **800***bd* into the inserted Knowledge Cell **800***hd*. The system may also create Connection **853***h***3** between Knowledge Cell **800***hc* and Knowledge Cell **800***hd* with occurrence count of 1 and weight calculated based on the occurrence count as previously described. The system may also update weights of other outgoing Connections **853** (one in this example) originating from Knowledge Cell **800***hc* as previously described. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***be* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Graph **530***b*. In the case that a substantially similar match is not found, the system may insert Knowledge Cell **800***he* into Graph **530***b* and copy Knowledge Cell **800***be* into the inserted Knowledge Cell **800***he*. The system may also create Connection **853***h***4** between Knowledge Cell **800***hd* and Knowledge Cell **800***he* with occurrence count of 1 and weight of 1. Applying any additional Knowledge Cells **800** from Knowledge Structuring Unit **520** onto Graph **530***b* follows similar logic or process as the above-described.

Referring to FIG. **23**, an embodiment of learning Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Collection of Sequences **530***c* is illustrated. Collection of Sequences **530***c* comprises the functionality for storing one or more Sequences **533**. Sequence **533** comprises the functionality for storing any number of Knowledge Cells **800**. For example, Knowledge Structuring Unit **520** structures or generates Knowledge Cells **800** and the system applies them onto Collection of Sequences **530***c*, thereby implementing learning Device's **98** operation in circumstances including objects with various properties. The system can perform collective Similarity Comparisons **125** of Knowledge Cells **800** from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Sequences **533** of Collection of Sequences **530***c* to find a Sequence **533** comprising Knowledge Cells **800** that are collectively substantially similar to the Knowledge Cells **800** from Knowledge Structuring Unit **520**. If Sequence **533** comprising such collectively substantially similar Knowledge Cells **800** is not found in Collection of Sequences **530***c*, the system may create a new Sequence **533** comprising the Knowledge Cells **800** from Knowledge Structuring Unit **520** and insert (i.e. copy, store, etc.) the new Sequence **533** into Collection of Sequences **530***c*. On the other hand, if Sequence **533** comprising collectively substantially similar Knowledge Cells **800** is found in Collection of Sequences **530***c*, the system may optionally omit inserting the Knowledge Cells **800** from Knowledge Structuring Unit **520** into Collection of Sequences **530***c* as inserting a similar Sequence **533** may not add much or any additional knowledge. This approach can save storage resources and limit the number of Knowledge Cells **800** that may later need to be processed or compared. In some aspects, a Sequence **533** may include Knowledge Cells **800** relating to a single operation of Device **98**. In other aspects, a Sequence **533** may include Knowledge Cells **800** relating to a part of an operation of Device **98**. In further aspects, one or more long Sequences **533** each including Knowledge Cells **800** of multiple operations of Device **98** can be utilized. In one example, Knowledge Cells **800** of all operations can be stored in a single long Sequence **533** in which case Collection of Sequences **530***c* as a separate element can be omitted. In another example, Knowledge Cells **800** of multiple operations can be included in a plurality of long Sequences **533** such as hourly, daily, weekly, monthly, yearly, or other periodic or other Sequences **533**. Similarity Comparisons **125** can be performed by traversing the one or more long Sequences **533** to find a match or substantially similar match. For instance, the system can perform collective Similarity Comparisons **125** of Knowledge Cells **800** from Knowledge Structuring Unit **520** with Knowledge Cells **800** in subsequences of a long Sequence **533** in incremental or other traversing pattern to find a subsequence comprising Knowledge Cells **800** that are collectively substantially similar to the Knowledge Cells **800** from Knowledge Structuring Unit **520**. The incremental traversing pattern may start from one end of a long Sequence **533** and move the comparison subsequence up or down one or any number of incremental Knowledge Cells **800** at a time. Other traversing patterns or methods can be employed such as starting from the middle of the Sequence **533** and subdividing the resulting sub-sequences in a recursive pattern, or any other traversing pattern or method. If a subsequence comprising collectively substantially similar Knowledge Cells **800** is not found in the long Sequence **533**, the system may concatenate or append the Knowledge Cells **800** from Knowledge Structuring Unit **520** to the long Sequence **533**. In further aspects, Connections **853** can optionally be used in Sequence **533** to connect Knowledge Cells **800**. For example, a Knowledge Cell **800** can be connected not only with a next Knowledge Cell **800** in the Sequence **533**, but also with any other Knowledge Cell **800** in the Sequence **533**, thereby creating alternate routes or shortcuts through the Sequence **533**. Any number of Connections **853** connecting any Knowledge Cells **800** can be utilized. Any of the previously described and/or other techniques for comparing,

inserting, updating, and/or other operations on Knowledge Cells **800**, Connections **853**, and/or other elements can similarly be utilized in Sequences **533** and/or Collection of Sequences **530***c*.

In some embodiments, various elements and/or techniques can be utilized in the aforementioned substantial similarity determinations with respect to collectively compared Knowledge Cells **800** and/or other elements. In some aspects, substantial similarity of collectively compared Knowledge Cells **800** can be determined based on similarities or similarity indexes of the individually compared Knowledge Cells **800**. In one example, an average of similarities or similarity indexes of individually compared Knowledge Cells **800** can be used to determine similarity of collectively compared Knowledge Cells **800**. In another example, a weighted average of similarities or similarity indexes of individually compared Knowledge Cells **800** can be used to determine similarity of collectively compared Knowledge Cells **800**. For instance, to affect the weighting of collective similarity, a higher weight or importance (i.e. importance index, etc.) can be assigned to the similarities or similarity indexes of some Knowledge Cells **800** and lower for other Knowledge Cells **800**. Any higher or lower weight or importance assignment can be implemented. In other aspects, any of the previously described or other thresholds for substantial similarity of individually compared elements can similarly be utilized for collectively compared elements. In one example, substantial similarity of collectively compared Knowledge Cells **800** can be achieved when their collective similarity or similarity index exceeds a similarity threshold. In another example, substantial similarity of collectively compared Knowledge Cells **800** can be achieved when at least a threshold number or percentage of Knowledge Cells **800** from the collectively compared Knowledge Cells **800** match or substantially match. Similarly, substantial similarity of collectively compared Knowledge Cells **800** can be achieved when a number or percentage of matching or substantially matching Knowledge Cells **800** from the collectively compared Knowledge Cells **800** exceeds a threshold. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. Similar elements and/or techniques as the aforementioned can be used for similarity determinations of other collectively compared elements such as Collections of Object Representations **525**, Object Representations **625**, Object Properties **630**, Instruction Sets **526**, Extra Info **527**, and/or others. Similarity determinations of collectively compared elements may include any features, functionalities, and embodiments of Similarity Comparison **125**, and vice versa.

Any of the previously described data structures or arrangements of Knowledge Cells **800** such as Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, and/or others can be used alone, or in combination with each other or with other elements, in alternate embodiments. In one example, a path in Neural Network **530***a* or Graph **530***b* may include its own separate sequence of Knowledge Cells **800** that are not interconnected with Knowledge Cells **800** in other paths. In another example, a part of a path in Neural Network **530***a* or Graph **530***b* may include a sequence of Knowledge Cells **800** interconnected with Knowledge Cells **800** in other paths, whereas, another part of the path may include its own separate sequence of Knowledge Cells **800** that are not interconnected with Knowledge Cells **800** in other paths. Any other combinations or arrangements of Knowledge Cells **800** can be implemented.

Referring to FIG. **24**, an embodiment of determining anticipatory Instruction Sets **526** from a single Knowledge Cell **800** is illustrated. Knowledge Cell **800** may be part of a Knowledgebase **530** (i.e. Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.) such as Collection of Knowledge Cells **530***d*. Decision-making Unit **540** comprises the functionality for anticipating or determining a device's operation in circumstances including objects with various properties. Decision-making Unit **540** comprises the functionality for anticipating or determining Instruction Sets **526** to be used or executed in Device's **98** autonomous operation. In some aspects, Instruction Sets **526** anticipated or determined to be used or executed in Device's **98** autonomous operation may be referred to as anticipatory Instruction Sets **526**, alternate Instruction Sets **526**, and/or other suitable name or reference. Therefore, these terms can be used interchangeably herein depending on context. Decision-making Unit **540** also comprises other disclosed functionalities.

In some aspects, Decision-making Unit **540** may anticipate or determine Instruction Sets **526** (i.e. anticipatory Instruction Sets **526**, etc.) for autonomous Device **98** operation by performing Similarity Comparisons **125** of incoming Collections of Object Representations **525** or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Knowledgebase **530** (i.e. Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.). A Knowledge Cell **800** includes knowledge (i.e. one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**, etc.) of how Device **98** operated in a circumstance including objects with various properties as previously described. When one or more Collections of Object Representations **525** representing objects with similar properties are received in the future, Decision-making Unit **540** can anticipate the Instruction Sets **526** (i.e. anticipatory Instruction Sets **526**, etc.) previously learned in a similar circumstance, thereby enabling autonomous Device **98** operation. In some aspects, Decision-making Unit **540** can perform Similarity Comparisons **125** of incoming Collections of Object Representations **525** from Object Processing Unit **93** with Collections of Object Representations **525** from Knowledge Cells **800** in Knowledgebase **530** (i.e. Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.). If one or more substantially similar Collections of Object Representations **525** or portions thereof are found in a Knowledge Cell **800** from Knowledgebase **530**, Instruction Sets **526** (i.e. anticipatory Instruction Sets **526**, etc.) for autonomous Device **98** operation can be anticipated in Instruction Sets **526** correlated with the one or more Collections of Object Representations **525** from the Knowledge Cell **800**. In some designs, subsequent one or more Instruction Sets **526** for autonomous Device **98** operation can be anticipated in Instruction Sets **526** correlated with subsequent Collections of Object Representations **525** from the Knowledge Cell **800** or other Knowledge Cells **800**, thereby anticipating not only current, but also additional future Instruction Sets **526**. Although, Extra Info **527** is not shown in this and/or other figures for clarity of illustration, it should be noted that any Collection of Object Representations **525**, Instruction Set **526**, and/or other element may include or be associated with Extra Info

527 and that Decision-making Unit **540** can utilize Extra Info **527** for enhanced decision making.

For example, Decision-making Unit **540** can perform Similarity Comparison **125** of Collection of Object Representations **525l1** or portions thereof from Object Processing Unit **93** with Collection of Object Representations **525*a*1** or portions thereof from Knowledge Cell **800*oa***. Collection of Object Representations **525*a*1** or portions thereof from Knowledge Cell **800*oa*** may be found substantially similar. Decision-making Unit **540** can anticipate Instruction Sets **526*a*1**-**526*a*3** correlated with Collection of Object Representations **525*a*1**, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparison **125** of Collection of Object Representations **525l2** or portions thereof from Object Processing Unit **93** with Collection of Object Representations **525*a*2** or portions thereof from Knowledge Cell **800*oa***. Collection of Object Representations **525*a*2** or portions thereof from Knowledge Cell **800*oa*** may be found substantially similar. Decision-making Unit **540** can anticipate Instruction Set **526*a*4** correlated with Collection of Object Representations **525*a*2**, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparison **125** of Collection of Object Representations **525l3** or portions thereof from Object Processing Unit **93** with Collection of Object Representations **525*a*3** or portions thereof from Knowledge Cell **800*oa***. Collection of Object Representations **525*a*3** or portions thereof from Knowledge Cell **800*oa*** may be found substantially similar. Decision-making Unit **540** may not anticipate any Instruction Sets **526** since none are correlated with Collection of Object Representations **525*a*3**. Decision-making Unit **540** can then perform Similarity Comparison **125** of Collection of Object Representations **525l4** or portions thereof from Object Processing Unit **93** with Collection of Object Representations **525*a*4** or portions thereof from Knowledge Cell **800*oa***. Collection of Object Representations **525*a*4** or portions thereof from Knowledge Cell **800*oa*** may not be found substantially similar. Decision-making Unit **540** can then perform Similarity Comparison **125** of Collection of Object Representations **525l5** or portions thereof from Object Processing Unit **93** with Collection of Object Representations **525*a*5** or portions thereof from Knowledge Cell **800*oa***. Collection of Object Representations **525*a*5** or portions thereof from Knowledge Cell **800*oa*** may not be found substantially similar. Decision-making Unit **540** can implement similar logic or process for any additional Collections of Object Representations **525** from Object Processing Unit **93**, and so on.

It should be understood that any of the described elements and/or techniques in the foregoing example can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells **800** or elements (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof would be affected accordingly. In one example, Extra Info **527** can be included in the Similarity Comparisons **125** as previously described. In another example, as history of incoming Collections of Object Representations **525** becomes available, Decision-making Unit **540** can perform collective Similarity Comparisons **125** of the history of Collections of Object Representations **525** or portions thereof from Object Processing Unit **93** with subsequences of Collections of Object Representations **525** or portions thereof from Knowledge Cell **800**. In a further example, the described comparisons in a single Knowledge Cell **800** may be performed on any number of Knowledge Cells **800** sequentially or in parallel. Parallel processors such as a plurality of Processors **11** or cores thereof can be utilized for such parallel processing. In a further example, various arrangements of Collections of Object Representations **525** and/or other elements in a Knowledge Cell **800** can be utilized as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to FIG. **25**, an embodiment of determining anticipatory Instruction Sets **526** by traversing a single Knowledge Cell **800** is illustrated. Knowledge Cell **800** may be part of a Knowledgebase **530** (i.e. Neural Network **530*a***, Graph **530*b***, Collection of Sequences **530*c***, Sequence **533**, Collection of Knowledge Cells **530*d***, etc.) such as Collection of Knowledge Cells **530*d***. For example, Decision-making Unit **540** can perform Similarity Comparison **125** of Collection of Object Representations **525l1** or portions thereof from Object Processing Unit **93** with Collection of Object Representations **525*a*1** or portions thereof from Knowledge Cell **800*oa***. Collection of Object Representations **525*a*1** or portions thereof from Knowledge Cell **800*oa*** may not be found substantially similar. Decision-making Unit **540** can then perform Similarity Comparison **125** of Collection of Object Representations **525l1** or portions thereof from Object Processing Unit **93** with Collection of Object Representations **525*a*2** or portions thereof from Knowledge Cell **800*oa***. Collection of Object Representations **525*a*2** or portions thereof from Knowledge Cell **800*oa*** may not be found substantially similar. Decision-making Unit **540** can then perform Similarity Comparison **125** of Collection of Object Representations **525l1** or portions thereof from Object Processing Unit **93** with Collection of Object Representations **525*a*3** or portions thereof from Knowledge Cell **800*oa***. Collection of Object Representations **525*a*3** or portions thereof from Knowledge Cell **800*oa*** may be found substantially similar. Decision-making Unit **540** may not anticipate any Instruction Sets **526** since none are correlated with Collection of Object Representations **525*a*3**. Decision-making Unit **540** can then perform Similarity Comparison **125** of Collection of Object Representations **525l2** or portions thereof from Object Processing Unit **93** with Collection of Object Representations **525*a*4** or portions thereof from Knowledge Cell **800*oa***. Collection of Object Representations **525*a*4** or portions thereof from Knowledge Cell **800*oa*** may be found substantially similar. Decision-making Unit **540** can anticipate Instruction Sets **526*a*5**-**526*a*6** correlated with Collection of Object Representations **525*a*4**, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparison **125** of Collection of Object Representations **525l3** or portions thereof from Object Processing Unit **93** with Collection of Object Representations **525*a*5** or portions thereof from Knowledge Cell **800*oa***. Collection of Object Representations **525*a*5** or portions thereof from Knowledge Cell **800*oa*** may be found substantially similar. Decision-making Unit **540** may not anticipate any Instruction Sets **526** since none are correlated with Collection of Object Representations **525*a*5**. Decision-making Unit **540** can implement similar logic or process for any additional Collections of Object Representations **525** from Object Processing Unit **93**, and so on.

It should be understood that any of the described elements and/or techniques in the foregoing example can be omitted, used in a different combination, or used in combination with

other elements and/or techniques, in which case the selection of Knowledge Cells **800** or elements (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof would be affected accordingly. In one example, Extra Info **527** can be included in the Similarity Comparisons **125** as previously described. In another example, as history of incoming Collections of Object Representations **525** becomes available, Decision-making Unit **540** can perform collective Similarity Comparisons **125** of the history of Collections of Object Representations **525** or portions thereof from Object Processing Unit **93** with subsequences of Collections of Object Representations **525** or portions thereof from Knowledge Cell **800**. In a further example, traversing may be performed in incremental traversing pattern such as starting from one end of Knowledge Cell **800** and moving the comparison subsequence up or down the list one or any number of incremental Collections of Object Representations **525** at a time. Other traversing patterns or methods can be employed such as starting from the middle of the Knowledge Cell **800** and subdividing the resulting subsequence in a recursive pattern, or any other traversing pattern or method. In a further example, the described traversing of a single Knowledge Cell **800** may be performed on any number of Knowledge Cells **800** sequentially or in parallel. Parallel processors such as a plurality of Processors **11** or cores thereof can be utilized for such parallel processing. In a further example, various arrangements of Collections of Object Representations **525** and/or other elements in a Knowledge Cell **800** can be utilized as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to FIG. **26**, an embodiment of determining anticipatory Instruction Sets **526** using collective similarity comparisons is illustrated. For example, Decision-making Unit **540** can perform Similarity Comparisons **125** of Collection of Object Representations **525l1** or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. Collection of Object Representations **525***c***1** or portions thereof from Knowledge Cell **800***rc* may be found substantially similar with highest similarity. Decision-making Unit **540** can anticipate any Instruction Sets **526** (not shown) correlated with Collection of Object Representations **525***c***1**, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform collective Similarity Comparisons **125** of Collections of Object Representations **525l1-525l2** or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. Collections of Object Representations **525***c***1-525***c***2** or portions thereof from Knowledge Cell **800***rc* may be found substantially similar with highest similarity. Decision-making Unit **540** can anticipate any Instruction Sets **526** (not shown) correlated with Collection of Object Representations **525***c***2**, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform collective Similarity Comparisons **125** of Collections of Object Representations **525l1-525l3** or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. Collections of Object Representations **525***d***1-525***d***3** or portions thereof from Knowledge Cell **800***rd* may be found substantially similar with highest similarity. Decision-making Unit **540** can anticipate any Instruction Sets **526** (not shown) correlated with Collection of Object Representations **525***d***3**, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform collective Similarity Comparisons **125** of Collections of Object Representations **525l1-525l4** or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. Collections of Object Representations **525***d***1-525***d***4** or portions thereof from Knowledge Cell **800***rd* may be found substantially similar with highest similarity. Decision-making Unit **540** can anticipate any Instruction Sets **526** (not shown) correlated with Collection of Object Representations **525***d***4**, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform collective Similarity Comparisons **125** of Collections of Object Representations **525l1-525l5** or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. Collections of Object Representations **525***d***1-525***d***5** or portions thereof from Knowledge Cell **800***rd* may be found substantially similar with highest similarity. Decision-making Unit **540** can anticipate any Instruction Sets **526** (not shown) correlated with Collection of Object Representations **525***d***5**, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can implement similar logic or process for any additional Collections of Object Representations **525** from Object Processing Unit **93**, and so on.

In some embodiments, various elements and/or techniques can be utilized in the aforementioned similarity determinations with respect to collectively compared Collections of Object Representations **525** and/or other elements. In some aspects, similarity of collectively compared Collections of Object Representations **525** can be determined based on similarities or similarity indexes of the individually compared Collections of Object Representations **525**. In one example, an average of similarities or similarity indexes of individually compared Collections of Object Representations **525** can be used to determine similarity of collectively compared Collections of Object Representations **525**. In another example, a weighted average of similarities or similarity indexes of individually compared Collections of Object Representations **525** can be used to determine similarity of collectively compared Collections of Object Representations **525**. For instance, to affect the weighting of collective similarity, a higher weight or importance (i.e. importance index, etc.) can be assigned to the similarities or similarity indexes of some (i.e. more substantive or larger, etc.) Collections of Object Representations **525** and lower for other (i.e. less substantive or smaller, etc.) Collections of Object Representations **525**. Any other higher or lower weight or importance assignment can be implemented. In other aspects, any of the previously described or other thresholds for substantial similarity of individually compared elements can be similarly utilized for collectively compared elements. In one example, substantial similarity of collectively compared Collections of Object Representations **525** can be achieved when their collective similarity or similarity index exceeds a similarity threshold. In another example, substantial similarity of collectively compared Collections of Object Representations **525** can be achieved when at least a threshold number or percentage of Collections of Object Representations **525** or portions thereof from

the collectively compared Collections of Object Representations **525** match or substantially match. Similarly, substantial similarity of collectively compared Collections of Object Representations **525** can be achieved when a number or percentage of matching or substantially matching Collections of Object Representations **525** or portions thereof from the collectively compared Collections of Object Representations **525** exceeds a threshold. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. Similar elements and/or techniques as the aforementioned can be used for similarity determinations of other collectively compared elements such as Object Representations **625**, Object Properties **630**, Instruction Sets **526**, Extra Info **527**, Knowledge Cells **800**, and/or others. Similarity determinations of collectively compared elements may include any features, functionalities, and embodiments of Similarity Comparison **125**, and vice versa.

It should be understood that any of the described elements and/or techniques in the foregoing example can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells **800** or elements (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info **527** in Similarity Comparisons **125**, traversing of Knowledge Cells **800** or other elements, using history of Collections of Object Representations **525** or Knowledge Cells **800** for collective Similarity Comparisons **125**, using various arrangements of Collections of Object Representations **525** and/or other elements in a Knowledge Cell **800**, and/or others can similarly be utilized in this example. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to FIG. **27**, an embodiment of determining anticipatory Instruction Sets **526** using Neural Network **530*a*** is illustrated. In some aspects, determining anticipatory Instruction Sets **526** using Neural Network **530*a*** may include selecting a path of Knowledge Cells **800** or elements (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof through Neural Network **530*a***. Decision-making Unit **540** can utilize various elements and/or techniques for selecting a path through Neural Network **530*a***. Although, these elements and/or techniques are described with respect to Neural Network **530*a*** below, they can similarly be used in any Knowledgebase **530** (i.e. Graph **530*b***, Collection of Sequences **530*c***, Sequence **533**, Collection of Knowledge Cells **530*d***, etc.) as applicable.

In some embodiments, Decision-making Unit **540** can utilize similarity index in selecting Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof in a path through Neural Network **530*a***. For instance, similarity index may indicate how well one Knowledge Cell **800** or portions thereof are matched with another Knowledge Cell **800** or portions thereof as previously described. In one example, Decision-making Unit **540** may select a Knowledge Cell **800** comprising one or more Collections of Object Representations **525** with highest similarity index even if Connection **853** pointing to that Knowledge Cell **800** has less than the highest weight. Therefore, similarity index or other such

element or parameter can override or disregard the weight of a Connection **853** or other element. In another example, Decision-making Unit **540** may select a Knowledge Cell **800** comprising one or more Collections of Object Representations **525** whose similarity index is higher than or equal to a weight of Connection **853** pointing to that Knowledge Cell **800**. In a further example, Decision-making Unit **540** may select a Knowledge Cell **800** comprising one or more Collections of Object Representations **525** whose similarity index is lower than or equal to a weight of Connection **853** pointing to that Knowledge Cell **800**. Similarity index can be set to be more, less, or equally important than a weight of a Connection **853**.

In some embodiments, Decision-making Unit **540** can utilize Connections **853** in selecting Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof in a path through Neural Network **530*a***. In some aspects, Decision-making Unit **540** can take into account weights of Connections **853** among the interconnected Knowledge Cells **800** in choosing from which Knowledge Cell **800** to compare one or more Collections of Object Representations **525** first, second, third, and so on. Specifically, for instance, Decision-making Unit **540** can perform Similarity Comparisons **125** with one or more Collections of Object Representations **525** from Knowledge Cell **800** pointed to by the highest weight Connection **853** first, Collections of Object Representations **525** from Knowledge Cell **800** pointed to by the second highest weight Connection **853** second, and so on. In other aspects, Decision-making Unit **540** can stop performing Similarity Comparisons **125** as soon as it finds one or more substantially similar Collections of Object Representations **525** in an interconnected Knowledge Cell **800**. In further aspects, Decision-making Unit **540** may only follow the highest weight Connection **853** to arrive at a Knowledge Cell **800** comprising one or more Collections of Object Representations **525** to be compared, thereby disregarding Connections **853** with less than the highest weight. In further aspects, Decision-making Unit **540** may ignore weights and/or other parameters of Connections **853**. In further aspects, Decision-making Unit **540** may ignore Connections **853**.

In some embodiments, Decision-making Unit **540** can utilize a bias to adjust similarity index, weight of a Connection **853**, and/or other element or parameter used in selecting Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof in a path through Neural Network **530*a***. In one example, Decision-making Unit **540** may select a Knowledge Cell **800** comprising one or more Collections of Object Representations **525** whose similarity index multiplied by or adjusted for a bias is higher than or equal to a weight of Connection **853** pointing to that Knowledge Cell **800**. In another example, Decision-making Unit **540** may select a Knowledge Cell **800** comprising one or more Collections of Object Representations **525** whose similarity index multiplied by or adjusted for a bias is lower than or equal to a weight of Connection **853** pointing to that Knowledge Cell **800**. In a further example, bias can be used to resolve deadlock situations where similarity index is equal to a weight of a Connection **853**. In some aspects, bias can be expressed in percentages such as 0.3 percent, 1.2 percent, 25.7 percent, 79.8 percent, 99.9 percent, 100.1 percent, 155.4 percent, 298.6 percent, 1105.5 percent, and so on. For example, a bias below 100 percent decreases an element or parameter to which it is applied, a bias equal to 100 percent does not change the element or parameter to which it is

applied, and a bias higher than 100 percent increases the element or parameter to which it is applied. In general, any amount of bias can be utilized depending on implementation. Bias can be applied to one or more of a weight of a Connection **853**, similarity index, any other element or parameter, and/or all or any combination of them. Also, different biases can be applied to each of a weight of a Connection **853**, similarity index, or any other element or parameter. For example, 30 percent bias can be applied to similarity index and 15 percent bias can be applied to a weight of a Connection **853**. Also, different biases can be applied to various Layers **854** of Neural Network **530***a*, and/or other disclosed elements. Bias can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input.

Any other element and/or technique can be utilized in selecting Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof in a path through Neural Network **530***a*.

In some embodiments, Neural Network **530***a* may include knowledge (i.e. interconnected Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**, etc.) of how Device **98** operated in circumstances including objects with various properties. In some aspects, determining anticipatory Instruction Sets **526** using Neural Network **530***a* may include selecting a path of Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof through Neural Network **530***a*. Individual and/or collective Similarity Comparisons **125** can be used to determine substantial similarity of the individually and/or collectively compared Collections of Object Representations **525** or portions thereof. Substantial similarity may be used primarily for selecting a path through Neural Network **530***a*, whereas, weight of any Connection **853** may be used secondarily or not at all.

For example, Decision-making Unit **540** can perform Similarity Comparisons **125** of Collections of Object Representations **525***a***1**-**525***an* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from one or more Knowledge Cells **800** in Layer **854***a* (or any other one or more Layers **854**, etc.). Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***ta* may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525***b***1**-**525***bn* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from one or more Knowledge Cells **800** in Layer **854***b* interconnected with Knowledge Cell **800***ta*. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***tb* may be found collectively substantially similar with highest similarity, thus, Decision-making Unit **540** may follow Connection **853***t***1** disregarding its less than highest weight. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Since Connection **853***t***2** is the only connection from Knowledge Cell **800***tb*, Decision-making Unit **540** may follow Connection **853***t***2** and perform Similarity Comparisons **125** of Collections of Object Representations **525***c***1**-**525***cn* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***tc* in Layer **854***c*. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***tc* may be found collectively substantially similar. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525***d***1**-**525***dn* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from one or more Knowledge Cells **800** in Layer **854***d* interconnected with Knowledge Cell **800***tc*. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***td* may be found collectively substantially similar with highest similarity, thus, Decision-making Unit **540** may follow Connection **853***t***3**. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525***e***1**-**525***en* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from one or more Knowledge Cells **800** in Layer **854***e* interconnected with Knowledge Cell **800***td*. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***te* may be found collectively substantially similar with highest similarity, thus, Decision-making Unit **540** may follow Connection **853***t***4**. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can implement similar logic or process for any additional Collections of Object Representations **525** from Object Processing Unit **93**, and so on.

The foregoing exemplary embodiment provides an example of utilizing a combination of collective Similarity Comparisons **125**, individual Similarity Comparisons **125**, Connections **853**, and/or other elements or techniques. It should be understood that any of these elements and/or techniques can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof in a path through Neural Network **530***a* would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info **527** in Similarity Comparisons **125**, traversing of Knowledge

Cells **800** or other elements, using history of Collections of Object Representations **525** or Knowledge Cells **800** for collective Similarity Comparisons **125**, using various arrangements of Collections of Object Representations **525** and/or other elements in a Knowledge Cell **800**, and/or others can similarly be utilized in this example. These elements and/or techniques can similarly be utilized in Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, and/or other data structures or arrangements. In some aspects, instead of anticipating Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525**, Decision-making Unit **540** can anticipate instruction Sets **526** correlated with substantially similar streams of Collections of Object Representations **525**. In other aspects, any time that substantial similarity or other similarity threshold is not achieved in compared Collections of Object Representations **525** or portions thereof from any of the Knowledge Cells **800**, Decision-making Unit **540** can decide to look for a substantially or otherwise similar Collections of Object Representations **525** or portions thereof in Knowledge Cells **800** elsewhere in Neural Network **530***a* such as in any Layer **854** subsequent to a current Layer **854**, in the first Layer **854**, in the entire Neural Network **530***a*, and/or others, even if such Knowledge Cell **800** may be unconnected with a prior Knowledge Cell **800**. It should be noted that any of Collections of Object Representations **525***a***1**-**525***an*, Collections of Object Representations **525***b***1**-**525***bn*, Collections of Object Representations **525***c***1**-**525***cn*, Collections of Object Representations **525***d***1**-**525***dn*, Collections of Object Representations **525***e***1**-**525***en*, etc. may include one Collection of Object Representations **525** or a stream of Collections of Object Representations **525**. It should also be noted that any Knowledge Cell **800** may include one Collection of Object Representations **525** or a stream of Collections of Object Representations **525** as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to FIG. **28**, an embodiment of determining anticipatory Instruction Sets **526** using Graph **530***b* is illustrated. Graph **530***b* may include knowledge (i.e. interconnected Knowledge Cells **800** comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**, etc.) of how Device **98** operated in circumstances including objects with various properties. In some aspects, determining anticipatory Instruction Sets **526** using Graph **530***b* may include selecting a path of Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof through Graph **530***b*. Individual and/or collective Similarity Comparisons **125** can be used to determine substantial similarity of the individually and/or collectively compared Collections of Object Representations **525** or portions thereof. Substantial similarity may be used primarily for selecting a path through Graph **530***b*, whereas, weight of any Connection **853** may be used secondarily or not at all.

For example, Decision-making Unit **540** can perform Similarity Comparisons **125** of Collections of Object Representations **525***a***1**-**525***an* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from one or more Knowledge Cells **800** in Graph **530***b*. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***ua* may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525***b***1**-**525***bn* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from one or more Knowledge Cells **800** in Graph **530***b* interconnected with Knowledge Cell **800***ua* by outgoing Connections **853**. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***ub* may be found collectively substantially similar with highest similarity, thus, Decision-making Unit **540** may follow Connection **853***u***1** disregarding its less than highest weight. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525***c***1**-**525***cn* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from one or more Knowledge Cells **800** in Graph **530***b* interconnected with Knowledge Cell **800***ub* by outgoing Connections **853**. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***uc* may be found collectively substantially similar with highest similarity, thus, Decision-making Unit **540** may follow Connection **853***u***2** disregarding its less than highest weight. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Since Connection **853***u***3** is the only connection from Knowledge Cell **800***uc*, Decision-making Unit **540** may follow Connection **853***u***3** and perform Similarity Comparisons **125** of Collections of Object Representations **525***d***1**-**525***dn* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***ud* in Graph **530***b*. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***ud* may be found collectively substantially similar. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525***e***1**-**525***en* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from one or more Knowledge Cells **800** in Graph **530***b* interconnected with Knowledge Cell **800***ud* by outgoing Connections **853**. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***ue* may be found collectively substantially similar with highest similarity, thus, Decision-making Unit **540** may follow

Connection **853***u***4**. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can implement similar logic or process for any additional Collections of Object Representations **525** from Object Processing Unit **93**, and so on.

The foregoing exemplary embodiment provides an example of utilizing a combination of collective Similarity Comparisons **125**, individual Similarity Comparisons **125**, Connections **853**, and/or other elements or techniques. It should be understood that any of these elements and/or techniques can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof in a path through Graph **530***b* would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info **527** in Similarity Comparisons **125**, traversing of Knowledge Cells **800** or other elements, using history of Collections of Object Representations **525** or Knowledge Cells **800** in collective Similarity Comparisons **125**, using various arrangements of Collections of Object Representations **525** and/or other elements in a Knowledge Cell **800**, and/or others can similarly be utilized in this example. These elements and/or techniques can similarly be utilized in Neural Network **530***a*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, and/or other data structures or arrangements. In some aspects, instead of anticipating Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525**, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially matching streams of Collections of Object Representations **525**. In other aspects, any time that substantial similarity or other similarity threshold is not achieved in compared Collections of Object Representations **525** or portions thereof of any of the Knowledge Cells **800**, Decision-making Unit **540** can decide to look for a substantially or otherwise similar Collections of Object Representations **525** or portions thereof in Knowledge Cells **800** elsewhere in Graph **530***b* even if such Knowledge Cell **800** may be unconnected with a prior Knowledge Cell **800**. It should be noted that any of Collections of Object Representations **525***a***1**-**525***an*, Collections of Object Representations **525***b***1**-**525***bn*, Collections of Object Representations **525***c***1**-**525***cn*, Collections of Object Representations **525***d***1**-**525***dn*, Collections of Object Representations **525***e***1**-**525***en*, etc. may include one Collection of Object Representations **525** or a stream of Collections of Object Representations **525**. It should also be noted that any Knowledge Cell **800** may include one Collection of Object Representations **525** or a stream of Collections of Object Representations **525** as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to FIG. **29**, an embodiment of determining anticipatory Instruction Sets **526** using Collection of Sequences **530***c* is illustrated. Collection of Sequences **530***c* may include knowledge (i.e. sequences of Knowledge Cells

800 comprising one or more Collections of Object Representations **525** correlated with any Instruction Sets **526** and/or Extra Info **527**, etc.) of how Device **98** operated in circumstances including objects with various properties. In some aspects, determining anticipatory Instruction Sets **526** for autonomous Device **98** operation using Collection of Sequences **530***c* may include selecting a Sequence **533** of Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof from Collection of Sequences **530***c*. Individual and/or collective Similarity Comparisons **125** can be used to determine substantial similarity of the individually and/or collectively compared Collections of Object Representations **525** or portions thereof.

For example, Decision-making Unit **540** can perform Similarity Comparisons **125** of Collections of Object Representations **525***a***1**-**525***an* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in one or more Sequences **533** of Collection of Sequences **530***c*. Collections of Object Representations **525** or portions thereof from Knowledge Cell **800***ca* in Sequence **533***wc* may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525***a***1**-**525***an* and **525***b***1**-**525***bn* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Sequences **533** of Collection of Sequences **530***c*. Collections of Object Representations **525** or portions thereof from Knowledge Cells **800***ca*-**800***cb* in Sequence **533***wc* may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525***a***1**-**525***an*, **525***b***1**-**525***bn*, and **525***c***1**-**525***cn* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Sequences **533** of Collection of Sequences **530***c*. Collections of Object Representations **525** or portions thereof from Knowledge Cells **800***da*-**800***dc* in Sequence **533***wd* may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525***a***1**-**525***an*, **525***b***1**-**525***bn*, **525***c***1**-**525***cn*, and **525***d***1**-**525***dn* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Sequences **533** of Collection of Sequences **530***c*. Collections of Object Representations

525 or portions thereof from Knowledge Cells **800***da*-**800***dd* in Sequence **533***wd* may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Collections of Object Representations **525***a***1**-**525***an*, **525***b***1**-**525***bn*, **525***c***1**-**525***cn*, **525***d***1**-**525***dn*, and **525***e***1**-**525***en* or portions thereof from Object Processing Unit **93** with Collections of Object Representations **525** or portions thereof from Knowledge Cells **800** in Sequences **533** of Collection of Sequences **530***c*. Collections of Object Representations **525** or portions thereof from Knowledge Cells **800***da*-**800***de* in Sequence **533***wd* may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can implement similar logic or process for any additional Collections of Object Representations **525** from Object Processing Unit **93**, and so on.

The foregoing exemplary embodiment provides an example of utilizing a combination of collective Similarity Comparisons **125**, individual Similarity Comparisons **125**, and/or other elements or techniques. It should be understood that any of these elements and/or techniques can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Sequence **533** of Knowledge Cells **800** or portions (i.e. Collections of Object Representations **525**, Instruction Sets **526**, etc.) thereof would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info **527** in Similarity Comparisons **125**, traversing of Knowledge Cells **800** or other elements, using history of Collections of Object Representations **525** or Knowledge Cells **800** in collective Similarity Comparisons **125**, using various arrangements of Collections of Object Representations **525** and/or other elements in a Knowledge Cell **800**, and/or others can similarly be utilized in this example. These elements and/or techniques can similarly be utilized in Neural Network **530***a*, Graph **530***b*, Collection of Knowledge Cells **530***d*, and/or other data structures or arrangements. In some aspects, instead of anticipating Instruction Sets **526** correlated with substantially similar individual Collections of Object Representations **525**, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially matching streams of Collections of Object Representations **525**. In other aspects, any time that substantial similarity or other similarity threshold is not achieved in compared Collections of Object Representations **525** or portions thereof from any of the Knowledge Cells **800**, Decision-making Unit **540** can decide to look for a substantially or otherwise similar Collections of Object Representations **525** or portions thereof in Knowledge Cells **800** elsewhere in Collection of Sequences **530***c* such as in different Sequences **533**. It should be noted that any of Collections of Object Representations **525***a***1**-**525***an*, Collections of Object Representations **525***b***1**-**525***bn*, Collections of Object Representations **525***c***1**-**525***cn*, Collections of Object Representations **525***d***1**-**525***dn*, Collections of Object Representations **525***e***1**-**525***en*, etc. may include one Collection of Object Representations **525** or a stream of Collections of Object Representations **525**. It should also be noted that any Knowledge Cell **800** may include one Collection of Object Representations **525** or a stream of Collections of Object Representations **525** as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring now to Modification Interface **130**. Modification Interface **130** comprises the functionality for modifying execution and/or functionality of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. Modification Interface **130** comprises the functionality for modifying execution and/or functionality of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element at runtime. Modification Interface **130** comprises the functionality for modifying execution and/or functionality of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element based on anticipatory Instruction Sets **526**. In one example, Modification Interface **130** comprises the functionality to access, modify, and/or perform other manipulations on runtime engine/environment, virtual machine, operating system, compiler, just-in-time (JIT) compiler, interpreter, translator, execution stack, file, object, data structure, and/or other computing system elements. In another example, Modification Interface **130** comprises the functionality to access, modify, and/or perform other manipulations on memory, storage, bus, interfaces, and/or other computing system elements. In a further example, Modification Interface **130** comprises the functionality to access, modify, and/or perform other manipulations on Processor **11** registers and/or other Processor **11** elements. In a further example, Modification Interface **130** comprises the functionality to access, modify, and/or perform other manipulations on inputs and/or outputs of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. In a further example, Modification Interface **130** comprises the functionality to access, create, delete, modify, and/or perform other manipulations on functions, methods, procedures, routines, subroutines, and/or other elements of Application Program **18**. In a further example, Modification Interface **130** comprises the functionality to access, create, delete, modify, and/or perform other manipulations on source code, bytecode, compiled, interpreted, or otherwise translated code, machine code, and/or other code. In a further example, Modification Interface **130** comprises the functionality to access, create, delete, modify, and/or perform other manipulations on values, variables, parameters, and/or other data or information. Modification Interface **130** comprises any features, functionalities, and embodiments of Acquisition Interface **120**, and vice versa. Modification Interface **130** also comprises other disclosed functionalities.

Modification Interface **130** can employ various techniques for modifying execution and/or functionality of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. In some aspects, some of the previously described techniques and/or tools can be utilized. Code instrumentation, for instance, may involve inserting additional code, overwriting or rewriting existing code, and/or branching to a separate segment of code in

Application Program **18** as previously described. For example, instrumented code may include the following:

```
Device1.moveLeft(23);
modifyApplication( );
```

In the above sample code, instrumented call to Modification Interface's **130** function (i.e. modifyApplication( ), etc.) can be placed after a function (i.e. Device1.moveLeft(23), etc.) of Application Program **18**. Similar call to an application modifying function can be placed after or before some or all functions/routines/subroutines, some or all lines of code, some or all statements, some or all instructions or instruction sets, some or all basic blocks, and/or some or all other code segments of Application Program **18**. One or more application modifying function calls can be placed anywhere in Application Program's **18** code and can be executed at any points in Application Program's **18** execution. The application modifying function (i.e. modifyApplication( ), etc.) may include Artificial Intelligence Unit **110**-determined anticipatory Instruction Sets **526** that can modify execution and/or functionality of Application Program **18**. In some embodiments, the previously described obtaining Application Program's **18** instruction sets, data, and/or other information as well as modifying execution and/or functionality of Application Program **18** can be implemented in a single function that performs both tasks (i.e. traceAndModifyApplication( ), etc.).

In some embodiments, various computing systems and/or platforms may provide native tools for modifying execution and/or functionality of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. Independent vendors may provide tools with similar functionalities that can be utilized across different platforms. These tools enable a wide range of techniques or capabilities such as instrumentation, self-modifying code capabilities, dynamic code capabilities, branching, code rewriting, code overwriting, hot swapping, accessing and/or modifying objects or data structures, accessing and/or modifying functions/routines/subroutines, accessing and/or modifying variable or parameter values, accessing and/or modifying processor registers, accessing and/or modifying inputs and/or outputs, providing runtime memory access, and/or other capabilities. One of ordinary skill in art will understand that, while all possible variations of the techniques for modifying execution and/or functionality of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element are too voluminous to describe, these techniques are within the scope of this disclosure.

In one example, modifying execution and/or functionality of Application Program **18** can be implemented through utilizing metaprogramming techniques, which include applications that can self-modify or that can create, modify, and/or manipulate other applications. Self-modifying code, dynamic code, reflection, and/or other techniques can be used to facilitate metaprogramming. In some aspects, metaprogramming is facilitated through a programming language's ability to access and manipulate the internals of the runtime engine directly or via an API. In other aspects, metaprogramming is facilitated through dynamic execution of expressions (i.e. anticipatory Instruction Sets **526**, etc.) that can be created and/or executed at runtime. In yet other aspects, metaprogramming is facilitated through application modification tools, which can perform modifications on an application regardless of whether the application's programming language enables metaprogramming capabilities. Some operating systems may protect an application loaded into memory by restricting access to the loaded application. This protection mechanism can be circumvented by utilizing operating system's, processor's, and/or other low level features or commands to unprotect the loaded application. For example, a self-modifying application may modify the in-memory image of itself. To do so, the application can obtain the in-memory address of its code. The application may then change the operating system's or platform's protection on this memory range allowing it to modify the code (i.e. insert anticipatory Instruction Sets **526**, etc.). In addition to a self-modifying application, one application can utilize similar technique to modify another application. Linux mprotect command or similar commends of other operating systems can be used to change protection (i.e. unprotect, etc.) for a region of memory, for example. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program **18** can be implemented through native capabilities of dynamic, interpreted, and/or scripting programming languages and/or platforms. Most of these languages and/or platforms can perform functionalities at runtime that static programming languages may perform during compilation. Dynamic, interpreted, and/or scripting languages provide native functionalities such as self-modification of code, dynamic code, extending the application, adding new code, extending objects and definitions, and/or other functionalities that can modify an application's execution and/or functionality at runtime. Examples of dynamic, interpreted, and/or scripting languages include Lisp, Perl, PHP, JavaScript, Ruby, Python, Smalltalk, Tcl, VBScript, and/or others. Similar functionalities can also be provided in languages such as Java, C, and/or others using reflection. Reflection includes the ability of an application to examine and modify the structure and behavior of the application at runtime. For example, JavaScript can modify its own code as it runs by utilizing Function object constructor as follows:

```
myFunc=new Function(arg1, arg2, argN, functionBody);
```

The sample code above causes a new function object to be created with the specified arguments and body. The body and/or arguments of the new function object may include new instruction sets (i.e. anticipatory Instruction Sets **526**, etc.). The new function can be invoked as any other function in the original code. In another example, JavaScript can utilize eval method that accepts a string of JavaScript statements (i.e. anticipatory Instruction Sets **526**, etc.) and execute them as if they were within the original code. An example of how eval method can be used to modify an application includes the following JavaScript code:

```
anticipatoryInstr='Device1.moveForward(27);';
if (anticipatoryInstr !=" " && anticipatoryInstr !=null)
{
eval(anticipatoryInstr);
}
```

In the sample code above, Artificial Intelligence Unit **110** may generate anticipatory Instruction Set **526** (i.e. 'Device1.moveForward(27)' for moving a Device1 forward 27 units, etc.) and save it in anticipatoryInstr variable, which eval method can then execute. Lisp is another example of dynamic, interpreted, and/or scripting language that includes similar capabilities as previously described JavaScript. For example, Lisp's compile command can create a function at runtime, eval command may parse and evaluate an expression at runtime, and exec command may execute a given instruction set (i.e. string, etc.) at runtime. In another example, dynamic as well as some non-dynamic languages may provide macros, which combine code introspection and/or eval capabilities. In some aspects, macros can access inner workings of the compiler, interpreter, virtual machine,

runtime environment/engine, and/or other components of the computing platform enabling the definition of language-like constructs and/or generation of a complete program or sections thereof. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program **18** can be implemented through dynamic code, dynamic class loading, reflection, and/or other native functionalities of a programming language or platform. In static applications or static programming, a class can be defined and/or loaded at compile time. Conversely, in dynamic applications or dynamic programming, a class can be loaded into a running environment at runtime. For example, Java Runtime Environment (JRE) may not require that all classes be loaded at compile time and class loading can occur when a class is first referenced at runtime. Dynamic class loading enables inclusion or injection of on-demand code and/or functionalities at runtime. System provided or custom class loaders may enable loading of classes into the running environment. Custom class loaders can be created to enable custom functionalities such as, for example, specifying a remote location from which a class can be loaded. In addition to dynamic loading of a pre-defined class, a class can also be created at runtime. In some aspects, a class source code can be created at runtime. A compiler such as javac, com.sun.tools.javac.Main, javax.tools, javax.tools.JavaCompiler, and/or other packages can then be utilized to compile the source code. Javac, com.sun.tools.javac.Main, javax.tools, javax.tools.JavaCompiler, and/or other packages may include an interface to invoke Java compiler from within a running application. A Java compiler may accept source code in a file, string, object (i.e. Java String, StringBuffer, CharSequence, etc.) and/or other source, and may generate Java bytecode (i.e. class file, etc.). Once compiled, a class loader can then load the compiled class into the running environment. In other aspects, a tool such as Javaassist (i.e. Java programming assistant) can be utilized to enable an application to create or modify a class at runtime. Javassist may include a Java library that provides functionalities to create and/or manipulate Java bytecode of an application as well as reflection capabilities. Javassist may provide source-level and bytecode-level APIs. Using the source-level API, a class can be created and/or modified using only source code, which Javassist may compile seamlessly on the fly. Javassist source-level API can therefore be used without knowledge of Java bytecode specification. Bytecode-level API enables creating and/or editing a class bytecode directly. In yet other aspects, similar functionalities to the aforementioned ones may be provided in tools such as Apache Commons BCEL (Byte Code Engineering Library), ObjectWeb ASM, CGLIB (Byte Code Generation Library), and/or others. Once a dynamic code or class is created and loaded, reflection in high-level programming languages such as Java and/or others can be used to manipulate or change the runtime behavior of an application. Examples of reflective programming languages and/or platforms include Java, JavaScript, Smalltalk, Lisp, Python, .NET Common Language Runtime (CLR), Tcl, Ruby, Perl, PHP, Scheme, PL/SQL, and/or others. Reflection can be used in an application to access, examine, modify, and/or manipulate a loaded class and/or its elements. Reflection in Java can be implemented by utilizing a reflection API such as java.lang.Reflect package. The reflection API provides functionalities such as, for example, loading or reloading a class, instantiating a new instance of a class, determining class and instance methods, invoking class and instance methods, accessing and manipulating a class, fields, methods and constructors, determining the modifiers for fields, methods, classes, and interfaces, and/or other functionalities. The above described dynamic code, dynamic class loading, reflection, and/or other functionalities are similarly provided in the .NET platform through its tools such as, for example, System.CodeDom.Compiler namespace, System.Reflection.Emit namespace, and/or other native or other .NET tools. Other platforms in addition to Java and .NET may provide similar tools and/or functionalities. In some designs, dynamic code, dynamic class loading, reflection, and/or other functionalities can be used to facilitate modification of an application by inserting or injecting instruction sets (i.e. anticipatory Instruction Sets **526**, etc.) into a running application. For example, an existing or dynamically created class comprising DCADO Unit **100** functionalities can be loaded into a running application through manual, automatic, or dynamic instrumentation. Once the class is created and loaded, an instance of DCADO Unit **100** class may be constructed. The instance of DCADO Unit **100** can then take or exert control of the application and/or implement alternate instruction sets (i.e. anticipatory Instruction Sets **526**, etc.) at any point in the application's execution. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program **18** can be implemented through independent tools that can be utilized across different platforms. Such tools provide instrumentation and/or other capabilities on more than one platform or computing system and may facilitate application modification or insertion of instruction sets (i.e. anticipatory Instruction Sets **526**, etc.). Examples of these tools include Pin, DynamoRIO, DynInst, Kprobes, KernInst, OpenPAT, DTrace, SystemTap, and/or others. In some aspects, Pin and/or any of its elements, methods, and/or techniques can be utilized for dynamic instrumentation. Pin can perform instrumentation by taking control of an application after it loads into memory. Pin may insert itself into the address space of an executing application enabling it to take control. Pin JIT compiler can then compile and implement alternate code (i.e. anticipatory Instruction Sets **526**, etc.). Pin provides an extensive API for instrumentation at several abstraction levels. Pin supports two modes of instrumentation, JIT mode and probe mode. JIT mode uses a just-in-time compiler to insert instrumentation and recompile program code while probe mode uses code trampolines for instrumentation. Pin was designed for architecture and operating system independence. In other aspects, KernInst and/or any of its elements, methods, and/or techniques can be utilized for dynamic instrumentation. KernInst includes an instrumentation framework designed for dynamically inserting code into a running kernel of an operating system. KernInst implements probe-based dynamic instrumentation where code can be inserted, changed, and/or removed at will. Kerninst API enables client tools to construct their own tools for dynamic kernel instrumentation to suit variety of purposes such as insertion of alternate instruction sets (i.e. anticipatory Instruction Sets **526**, etc.). Client tools can communicate with KernInst over a network (i.e. internet, wireless network, LAW, WAN, etc). Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program **18** can be implemented through utilizing operating system's native tools or capa-

bilities such as Unix ptrace command. Ptrace includes a system call that may enable one process to control another allowing the controller to inspect and manipulate the internal state of its target. Ptrace can be used to modify a running application such as modify an application with alternate instruction sets (i.e. anticipatory Instruction Sets **526**, etc.). By attaching to an application using the ptrace call, the controlling application can gain extensive control over the operation of its target. This may include manipulation of its instruction sets, execution path, file descriptors, memory, registers, and/or other components. Ptrace can single-step through the target's code, observe and intercept system calls and their results, manipulate the target's signal handlers, receive and send signals on the target's behalf, and/or perform other operations within the target application. Ptrace's ability to write into the target application's memory space enables the controller to modify the running code of the target application. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program **18** can be implemented through utilizing just-in-time (JIT) compiling. JIT compilation (also known as dynamic translation, dynamic compilation, etc.) includes compilation performed during an application's execution (i.e. runtime, etc.). A code can be compiled when it is about to be executed, and it may be cached and reused later without the need for additional compilation. In some aspects, a JIT compiler can convert source code or byte code into machine code. In other aspects, a JIT compiler can convert source code into byte code. JIT compiling may be performed directly in memory. For example, JIT compiler can output machine code directly into memory and immediately execute it. Platforms such as Java, .NET, and/or others may implement JIT compilation as their native functionality. Platform independent tools for custom system design may include JIT compilation functionalities as well. In some aspects, JIT compilation includes redirecting application's execution to a JIT compiler from a specific entry point. For example, Pin can insert its JIT compiler into the address space of an application. Once execution is redirected to it, JIT compiler may receive alternate instruction sets (i.e. anticipatory Instruction Sets **526**, etc.) immediately before their compilation. The JIT compiled instruction sets can be stored in memory or another repository from where they may be retrieved and executed. Alternatively, for example, JIT compiler can create a copy of the original application code or a segment thereof, and insert alternate code (i.e. anticipatory Instruction Sets **526**, etc.) before compiling the modified code copy. In some aspects, JIT compiler may include a specialized memory such as fast cache memory dedicated to JIT compiler functionalities from which the modified code can be fetched rapidly. JIT compilation and/or any compilation in general may include compilation, interpretation, or other translation into machine code, bytecode, and/or other formats or types of code. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program **18** can be implemented through dynamic recompilation. Dynamic recompilation includes recompiling an application or part thereof during execution. An application can be modified with alternate features or instruction sets that may take effect after recompilation. Dynamic recompilation may be practical in various types of applications including object oriented, event driven, forms based, and/or other applications. In a typical windows-based application, most of the action after initial startup occurs in response to user or system events such as moving the mouse, selecting a menu option, typing text, running a scheduled task, making a network connection, and/or other events when an event handler is called to perform an operation appropriate for the event. Generally, when no events are being generated, the application is idle. For example, when an event occurs and an appropriate event handler is called, instrumentation can be implemented in the application's source code to insert alternate instruction sets (i.e. anticipatory Instruction Sets **526**, etc.) at which point the modified source code can be recompiled and/or executed. In some aspects, the state of the application can be saved before recompiling its modified source code so that the application may continue from its prior state. Saving the application's state can be achieved by saving its variables, data structures, objects, location of its current instruction, and/or other necessary information in environmental variables, memory, or other repositories where they can be accessed once the application is recompiled. In other aspects, application's variables, data structures, objects, address of its current instruction, and/or other necessary information can be saved in a repository such as file, database, or other repository accessible to the application after recompilation of its source code. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program **18** can be implemented through modifying or redirecting Application Program's **18** execution path. Generally, an application can be loaded into memory and the flow of execution proceeds from one instruction set to the next until the end of the application. An application may include a branching mechanism that can be driven by keyboard or other input devices, system events, and/or other computing system components or events that may impact the execution path. The execution path can also be altered by an external application through acquiring control of execution and/or redirecting execution to a function, routine/subroutine, or an alternate code segment at any point in the application's execution. A branch, jump, or other mechanism can be utilized to implement the redirected execution. For example, a jump instruction can be inserted at a specific point in an application's execution to redirect execution to an alternate code segment. A jump instruction set may include, for example, an unconditional branch, which always results in branching, or a conditional branch, which may or may not result in branching depending on a condition. When executing an application, a computer may fetch and execute instruction sets in sequence until it encounters a branch instruction set. If the instruction set is an unconditional branch, or it is conditional and the condition is satisfied, the computer may fetch its next instruction set from a different instruction set sequence or code segment as specified by the branch instruction set. After the execution of the alternate code segment, control may be redirected back to the original jump point or to another point in the application. For example, modifying an application can be implemented by redirecting execution of an application to alternate instruction sets (i.e. anticipatory Instruction Sets **526**, etc.). Alternate instruction sets can be pre-compiled, pre-interpreted, or otherwise pre-translated and ready for execution. Alternate instruction sets can also be JIT compiled, JIT interpreted, or otherwise JIT translated before

execution. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program **18** can be implemented through assembly language. Assembly language instructions may be directly related with the architecture's machine instructions as previously described. Assembly language can, therefore, be a powerful tool for implementing direct hardware (i.e. processor registers, memory, etc.) access and manipulations as well as access and manipulations of specialized processor features or instructions. Assembly language can also be a powerful tool for implementing low-level embedded systems, real-time systems, interrupt handlers, self or dynamically modifying code, and/or other applications. Specifically, for instance, self or dynamically modifying code that can be used to facilitate modifying of an application can be seamlessly implemented using assembly language. For example, using assembly language, instruction sets can be dynamically created and loaded into memory similar to the ones that a compiler may generate. Furthermore, using assembly language, memory space of a loaded application can be accessed to modify (including rewrite, overwrite, etc.) original instruction sets or to insert jumps or branches to alternate code elsewhere in memory. Some operating systems may implement protection from changes to applications loaded into memory. Operating system's, processor's, or other low level features or commands can be used to unprotect the protected locations in memory before the change as previously described. Alternatively, a pointer that may reside in a memory location where it could be readily altered can be utilized where the pointer may reference alternate code. In one example, assembly language can be utilized to write alternate code (i.e. anticipatory Instruction Sets **526**, etc.) into a location in memory outside a running application's memory space. Assembly language can then be utilized to redirect the application's execution to the alternate code by inserting a jump or branch into the application's in-memory code, by redirecting program counter, or by other technique. In another example, assembly language can be utilized to overwrite or rewrite the entire or part of an application's in-memory code with alternate code. In some aspects, high-level programming languages can call an external assembly language program to facilitate application modification as previously described. In yet other aspects, relatively low-level programming languages such as C may allow embedding assembly language directly in their source code such as, for example, using asm keyword of C. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program **18** can be implemented through binary rewriting. Binary rewriting tools and/or techniques may modify an application's executable. In some aspects, modification can be minor such as in the case of optimization where the original executable's functionality is kept. In other aspects, modification may change the application's functionality such as by inserting alternate code (i.e. anticipatory Instruction Sets **526**, etc.). Examples of binary rewriting tools include SecondWrite, ATOM, DynamoRIO, Purify, Pin, EEL, DynInst, PLTO, and/or others. Binary rewriting may include disassembly, analysis, and/or modification of target application. Since binary rewriting works directly on machine code executable, it is independent of source language, compiler, virtual machine (if one is utilized), and/or other higher level abstraction layers. Also, binary rewriting tools can perform application modifications without access to original source code. Binary rewriting tools include static rewriters, dynamic rewriters, minimally-invasive rewriters, and/or others. Static binary rewriters can modify an executable when the executable is not in use (i.e. not running). The rewritten executable may then be executed including any new or modified functionality. Dynamic binary rewriters can modify an executable during its execution, thereby enabling modification of an application's functionality at runtime. In some aspects, dynamic rewriters can be used for instrumentation or selective modifications such as insertion of alternate code (i.e. anticipatory Instruction Sets **526**, etc.), and/or for other runtime transformations or modifications. For example, some dynamic rewriters can be configured to intercept an application's execution at indirect control transfers and insert instrumentation or other application modifying code. Minimally-invasive rewriters may keep the original machine code to the greatest extent possible. They support limited modifications such as insertion of jumps into and out of instrumented code. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

Referring to FIG. **30**, in a further example, modifying execution and/or functionality of Processor **11** can be implemented through modification of processor registers, memory, or other computing system components. In some aspects, modifying execution and/or functionality of Processor **11** can be implemented by redirecting Processor's **11** execution to alternate instruction sets (i.e. anticipatory Instruction Sets **526**, etc.). In one example, Program Counter **211** may hold or point to a memory address of the next instruction set that will be executed by Processor **11**. Artificial Intelligence Unit **110** may generate anticipatory Instruction Sets **526** and store them in Memory **12** as previously described. Modification Interface **130** may then change Program Counter **211** to point to the location in Memory **12** where anticipatory Instruction Sets **526** are stored. The anticipatory Instruction Sets **526** can then be fetched from the location in Memory **12** pointed to by the modified Program Counter **211** and loaded into Instruction Register **212** for decoding and execution. Once anticipatory Instruction Sets **526** are executed, Modification Interface **130** may change Program Counter **211** to point to the last instruction set before the redirection or to any other instruction set. In other aspects, anticipatory Instruction Sets **526** can be loaded directly into Instruction Register **212**. As previously described, examples of other processor or computing system components that can be used during an instruction cycle include memory address register (MAR), memory data register (MDR), data registers, address registers, general purpose registers (GPRs), conditional registers, floating point registers (FPRs), constant registers, special purpose registers, machine-specific registers, Register Array **214**, Arithmetic Logic Unit **215**, control unit, and/or other circuits or components. Any of the aforementioned processor registers, memory, or other computing system components can be accessed and/or modified to facilitate the disclosed functionalities. In some embodiments, processor interrupt may be issued to facilitate such access and/or modification. In some designs, modifying execution and/or functionality of Processor **11** can be implemented in a program, combination of programs and hardware, or purely hardware system. Dedicated hardware may be built to perform modifying execution and/or functionality of Processor **11** with marginal or no impact to computing overhead. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

Referring to FIGS. **31**A-**31**B, in a further example, modifying execution and/or functionality of Logic Circuit **250** can be implemented through modification of inputs and/or outputs of Logic Circuit **250**. While Processor **11** includes any type of logic circuit, Logic Circuit **250** is described separately herein to offer additional detail on its functioning. Logic Circuit **250** comprises the functionality for performing logic operations using the circuit's inputs and producing outputs based on the logic operations performed as previously described. In one example, Logic Circuit **250** may perform some logic operations using four input values and produce two output values. Modifying execution and/or functionality of Logic Circuit **250** can be implemented by replacing its input values with anticipatory input values (i.e. anticipatory Instruction Sets **526**, etc.). Artificial Intelligence Unit **110** may generate anticipatory input values as previously described. Modification Interface **130** can then transmit the anticipatory input values to Logic Circuit **250** through the four hardwired connections as shown in FIG. **31**A. Modification Interface **130** may use Switches **251** to prevent delivery of any input values that may be sent to Logic Circuit **250** from its usual input source. As such, DCADO Unit **100** may cause Logic Circuit **250** to perform its logic operations using the four anticipatory input values, thereby implementing autonomous Device **98** operation. In another example, Logic Circuit **250** may perform some logic operations using four input values and produce two output values. Modifying execution and/or functionality of Logic Circuit **250** can be implemented by replacing its output values with anticipatory output values (i.e. anticipatory Instruction Sets **526**, etc.). Artificial Intelligence Unit **110** may generate anticipatory output values (i.e. anticipatory Instruction Sets **526**, etc.) as previously described. Modification Interface **130** can then transmit the anticipatory output values through the two hardwired connections as shown in FIG. **31**B. Modification Interface **130** may use Switches **251** to prevent delivery of any output values that may be sent by Logic Circuit **250**. As such, DCADO Unit **100** may bypass Logic Circuit **250** and transmit the two anticipatory output values to downstream elements, thereby implementing autonomous Device **98** operation. In a further example, instead of or in addition to modifying input and/or output values of Logic Circuit **250**, the execution and/or functionality of Logic Circuit **250** may be modified by modifying values or signals in one or more Logic Circuit's **250** internal components such as registers, memories, buses, and/or others (i.e. similar to the previously described modifying of Processor **11** components, etc.). In some designs, modifying execution and/or functionality of Logic Circuit **250** can be implemented in a program, combination of programs and hardware, or purely hardware system. Dedicated hardware may be built to perform modifying execution and/or functionality of Logic Circuit **250** with marginal or no impact to computing overhead. Any of the elements and/or techniques for modifying execution and/or functionality of Logic Circuit **250** can similarly be implemented with Processor **11** and/or other processing elements.

In some embodiments, DCADO Unit **100** may directly modify the functionality of an actuator (previously described, not shown). For example, Logic Circuit **250** or other processing element may control an actuator that enables Device **98** to perform mechanical, physical, and/or other operations. An actuator may receive one or more input values or control signals from Logic Circuit **250** or other processing element directing the actuator to perform specific operations. Modifying functionality of an actuator can be implemented by replacing its input values with anticipatory input values (i.e. anticipatory Instruction Sets **526**, etc.) as previously described with respect to replacing input values of Logic Circuit **250**. Specifically, for instance, Artificial Intelligence Unit **110** may generate anticipatory input values as previously described. Modification Interface **130** can then transmit the anticipatory input values to the actuator. Modification Interface **130** may use Switches **251** to prevent delivery of any input values that may be sent to the actuator from its usual input source. As such, DCADO Unit **100** may cause the actuator to perform its operations using the anticipatory input values, thereby implementing autonomous Device **98** operation.

One of ordinary skill in art will recognize that FIGS. **31**A-**31**B depict one of many implementations of Logic Circuit **250** and that any number of input and/or output values can be utilized in alternate implementations. One of ordinary skill in art will also recognize that Logic Circuit **250** may include any number and/or combination of logic components to implement any logic operations.

Other additional techniques or elements can be utilized as needed for modifying execution and/or functionality of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing elements, or some of the disclosed techniques or elements can be excluded, or a combination thereof can be utilized in alternate embodiments.

Referring to FIG. **32**, the illustration shows an embodiment of a method **9100** for learning and/or using a device's circumstances for autonomous device operation. In some aspects, the method can be used on a computing enabled device or system to enable learning of a device's operation in circumstances including objects with various properties and enable autonomous device operation in similar circumstances. Method **9100** may include any action or operation of any of the disclosed methods such as method **9200**, **9300**, **9400**, **9500**, **9600**, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method **9100**.

At step **9105**, a first collection of object representations is received. A collection of object representations (i.e. Collection of Object Representations **525**, etc.) may include one or more object representations (i.e. Object Representations **625**, etc.), object properties (i.e. Object Properties **630**, etc.), and/or other elements or information. An object representation may include an electronic representation of an object (i.e. Object **615**, etc.) detected in a device's surrounding. In some aspects, a collection of object representations may include one or more object representations, object properties, and/or other elements or information detected in a device's (i.e. Device's **98**, etc.) surrounding at a particular time. A collection of object representations may, therefore, include knowledge (i.e. unit of knowledge, etc.) of a device's circumstances including objects with various properties at a particular time. In some designs, a collection of object representations may include or be associated with a time stamp (not shown), order (not shown), or other time related information. In some embodiments, a collection of object representations may include or be substituted with a stream of collections of object representations, and vice versa. Therefore, the terms collection of object representations and stream of collections of object representations may be used interchangeably herein depending on context. A stream of collections of object representations may include one collection of object representations or a group, sequence, or other plurality of collections of object representations. In some aspects, a stream of collections of object

representations may include one or more collections of object representations, and/or other elements or information detected in a device's surrounding over time. A stream of collections of object representations may, therefore, include knowledge (i.e. unit of knowledge, etc.) of a device's circumstances including objects with various properties over time. As circumstances including objects with various properties in a device's surrounding change (i.e. objects and/or their properties change, move, act, transform, etc.) over time, this change may be captured in a stream of collections of object representations. In some designs, each collection of object representations in a stream may include or be associated with the aforementioned time stamp, order, or other time related information. Examples of objects include biological objects (i.e. persons, animals, vegetation, etc.), nature objects (i.e. rocks, bodies of water, etc.), manmade objects (i.e. buildings, streets, ground/aerial/aquatic vehicles, etc.), and/or others. In some aspects, any part of an object may be detected as an object itself. For instance, instead of or in addition to detecting a vehicle as an object, a wheel and/or other parts of the vehicle may be detected as objects. In general, an object may include any object or part thereof that can be detected. Examples of object properties include existence of an object, type of an object (i.e. person, cat, vehicle, building, street, tree, rock, etc.), identity of an object (i.e. name, identifier, etc.), distance of an object, bearing/angle of an object, location of an object (i.e. distance and bearing/angle from a known point, object coordinates, etc.), shape/size of an object (i.e. height, width, depth, computer model, point cloud, etc.), activity of an object (i.e. motion, gestures, etc.), and/or other properties of an object. In general, an object property may include any attribute of an object (i.e. existence of an object, type of an object, identity of an object, shape/size of an object, etc.), any relationship of an object with the device, other objects, or the environment (i.e. distance of an object, bearing/angle of an object, friend/foe relationship, etc.), and/or other information related to an object. Objects and/or their properties can be detected by one or more sensors (i.e. Sensors **92**, etc.) and/or an object processing unit (i.e. Object Processing Unit **93**, etc.). A sensor may obtain or detect information about its environment. As such, one or more sensors can be used to detect objects and/or their properties in a device's surrounding. In some designs, a sensor may be part of a device whose circumstances are being used for DCADO functionalities. In other designs, a sensor may be part of a remote device whose circumstances are being used for DCADO functionalities. Examples of a sensor include a camera (i.e. Camera **92***a*, etc.), a microphone (i.e. Microphone **92***b*, etc.), a lidar (i.e. Lidar **92***c*, etc.), a radar (i.e. Radar **92***d*, etc.), a sonar (i.e. Sonar **92***e*, etc.), and/or others. An object processing unit may process output from a sensor to obtain information of interest. As such, an object processing unit can be used to process output from a sensor to detect objects and/or their properties in a device's surrounding. In some aspects, an object processing unit may create or generate a collection of object representations. In other aspects, an object processing unit may create or generate a stream of collections of object representations. An object processing unit may include a picture recognizer (i.e. Picture Recognizer **94***a*, etc.), a sound recognizer (i.e. Sound Recognizer **94***b*, etc.), a lidar processing unit (i.e. Lidar Processing Unit **94***c*, etc.), a radar processing unit (i.e. Radar Processing Unit **94***d*, etc.), a sonar processing unit (i.e. Sonar Processing Unit **94***e*, etc.), and/or other elements or functionalities. In general, an object processing unit may include any signal processing element or technique known in art as applicable. In some implementations, an object processing unit and/or any of its elements or functionalities can be included in sensor and/or other elements. Receiving comprises any action or operation by or for a Collection of Object Representations **525**, stream of Collections of Object Representations **525**, Object Representation **625**, Object Property **630**, Sensor **92**, Camera **92***a*, Microphone **92***b*, Lidar **92***c*, Radar **92***d*, Sonar **92***e*, Object Processing Unit **93**, Picture Recognizer **94***a*, Sound Recognizer **94***b*, Lidar Processing Unit **94***c*, Radar Processing Unit **94***d*, Sonar Processing Unit **94***e*, and/or other disclosed elements.

At step **9110**, a first one or more instruction sets for operating a device are received. In some embodiments, an instruction set (i.e. Instruction Set **526**, etc.) may be used or executed by a processor (i.e. Processor **11**, etc.) in operating a device. In other embodiments, an instruction set may be part of an application program (i.e. Application Program **18**, etc.) used in operating a device. For example, the application can run or execute on one or more processors or other processing elements. In further embodiments, an instruction set may be used or executed by a logic circuit (i.e. Logic Circuit **250**, etc.) in operating a device. For example, such instruction set may be or include one or more inputs into or outputs from a logic circuit. In further embodiments, an instruction set may be used by an actuator in operating a device. For example, such instruction set may be or include one or more inputs into an actuator. Operating a device includes performing or causing any operations on/by/with the device. In some designs, an instruction set can be received from a processor, application program, logic circuit, and/or other processing element as the instruction set is being used or executed. In other aspects, an instruction set can be received from a processor, application program, logic circuit, and/or other processing element after the instruction set is used or executed. In further aspects, an instruction set can be received from a processor, application program, logic circuit, and/or other processing element before the instruction set has been used or executed. In further aspects, an instruction set can be received from a running processor, running application program, running logic circuit, and/or other running processing element. As such, an instruction set can be received at runtime. In other designs, an instruction set can be received from an actuator. In further designs, an instruction set can be received from memory (i.e. Memory **12**, etc.), hard drive, or any other storage element or repository. In further designs, an instruction set can be received over a network such as Internet, local area network, wireless network, and/or other network. In further designs, an instruction set can be received by an interface (i.e. Acquisition Interface **120**, etc.) configured to obtain instruction sets from a processor, application program, logic circuit, actuator, and/or other element. In general, an instruction set can be received by any element of the system. One or more instruction sets may temporally correspond to a collection of object representations. In some aspects, an instruction set that temporally corresponds to a collection of object representations includes an instruction set used or executed at the time of generating the collection of object representations. In other aspects, an instruction set that temporally corresponds to a collection of object representations includes an instruction set used or executed within a certain time period before and/or after generating the collection of object representations. Any time period can be utilized depending on implementation. In further aspects, an instruction set that temporally corresponds to a collection of object representations includes an instruction set used or executed from the time of generating the collection of object representations to the

time of generating a next collection of object representations. In further aspects, an instruction set that temporally corresponds to a collection of object representations includes an instruction set used or executed from the time of generating a preceding collection of object representations to the time of generating the collection of object representations. Any other temporal relationship or correspondence between collections of object representations and correlated instruction sets can be implemented. In general, one or more instruction sets that temporally correspond to a collection of object representations enable structuring knowledge of a device's operation at or around the time of generating the collection of object representations. Such functionality enables spontaneous or seamless learning of a device's operation in circumstances including objects with various properties as the device is operated in real life situations. In some embodiments, an instruction set may include one or more commands, keywords, symbols (i.e. parentheses, brackets, commas, semicolons, etc.), instructions, operators (i.e. =, <, >, etc.), variables, values, objects, data structures, functions (i.e. Function1( ), FIRST( ) MIN( ) SQRT( ) etc.), parameters, states, signals, inputs, outputs, references thereto, and/or other components. In other embodiments, an instruction set may include source code, bytecode, intermediate code, compiled, interpreted, or otherwise translated code, runtime code, assembly code, machine code, and/or any other computer code. In further embodiments, an instruction set may include one or more inputs into and/or outputs from a logic circuit. In further embodiments, an instruction set may include one or more inputs into an actuator. Receiving comprises any action or operation by or for an Acquisition Interface **120**, Instruction Set **526**, and/or other disclosed elements.

At step **9115**, the first collection of object representations is correlated with the first one or more instruction sets for operating the device. In some aspects, individual collections of object representations can be correlated with one or more instruction sets. In other aspects, streams of collections of object representations can be correlated with one or more instruction sets. In further aspects, individual collections of object representations or streams of collections of object representations can be correlated with the aforementioned temporally corresponding instruction sets. In further aspects, a collection of object representations or stream of collections of object representations may not be correlated with any instruction sets. Correlating may include structuring or generating a knowledge cell (i.e.

Knowledge Cell **800**, etc.) and storing one or more collections of object representations correlated with any instruction sets into the knowledge cell. Therefore, a knowledge cell may include any data structure or arrangement that can facilitate such storing. A knowledge cell includes knowledge (i.e. unit of knowledge, etc.) of how a device operated in a circumstance including objects with various properties. In some designs, extra information (i.e. Extra Info **527**, etc.) may optionally be used to facilitate enhanced comparisons or decision making in autonomous device operation where applicable. Therefore, any collection of object representations, instruction set, and/or other element may include or be correlated with extra information. Extra information may include any information useful in comparisons or decision making performed in autonomous device operation. Examples of extra information include time information, location information, computed information, contextual information, and/or other information. Correlating may be omitted where learning of a device's operation in circumstances including objects with various properties is not implemented. Correlating comprises any action or operation by or for a Knowledge Structuring Unit **520**, Knowledge Cell **800**, and/or other disclosed elements.

At step **9120**, the first collection of object representations correlated with the first one or more instruction sets for operating the device are stored. A collection of object representations correlated with one or more instruction sets may be part of a stored plurality of collections of object representations correlated with one or more instruction sets. Collections of object representations correlated with any instruction sets can be stored in a memory unit or other repository. The aforementioned knowledge cells comprising collections of object representations correlated with any instruction sets can be used in/as neurons, nodes, vertices, or other elements in any of the data structures or arrangements (i.e. neural networks, graphs, sequences, collection of knowledge cells, etc.) used for storing the knowledge of a device's operation in circumstances including objects with various properties. Knowledge cells may be connected, interrelated, or interlinked into knowledge structures using statistical, artificial intelligence, machine learning, and/or other models or techniques. Such interconnected or interrelated knowledge cells can be used for enabling autonomous device operation. The interconnected or interrelated knowledge cells may be stored or organized into a knowledgebase (i.e. Knowledgebase **530**, etc.). In some embodiments, knowledgebase may be or include a neural network (i.e. Neural Network **530***a*, etc.). In other embodiments, knowledgebase may be or include a graph (i.e. Graph **530***b*, etc.). In further embodiments, knowledgebase may be or include a collection of sequences (i.e. Collection of Sequences **530***c*, etc.). In further embodiments, knowledgebase may be or include a sequence (i.e. Sequence **533**, etc.). In further embodiments, knowledgebase may be or include a collection of knowledge cells (i.e. Collection of Knowledge Cells **530***d*, etc.). In general, knowledgebase may be or include any data structure or arrangement, and/or repository capable of storing the knowledge of a device's operation in circumstances including objects with various properties. Knowledgebase may also include or be substituted with various artificial intelligence methods, systems, and/or models for knowledge structuring, storing, and/or representation such as deep learning, supervised learning, unsupervised learning, neural networks (i.e. convolutional neural network, recurrent neural network, deep neural network, etc.), search-based, logic and/or fuzzy logic-based, optimization-based, tree/graph/other data structure-based, hierarchical, symbolic and/or sub-symbolic, evolutionary, genetic, multi-agent, deterministic, probabilistic, statistical, and/or other methods, systems, and/or models. Storing may be omitted where learning of a device's operation in circumstances including objects with various properties is not implemented. Storing comprises any action or operation by or for a Knowledgebase **530**, Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, Knowledge Cell **800**, Node **852**, Layer **854**, Connection **853**, Similarity Comparison **125**, and/or other disclosed elements.

At step **9125**, a new collection of object representations is received. Step **9125** may include any action or operation described in Step **9105** as applicable.

At step **9130**, the new collection of object representations is compared with the first collection of object representations. Comparing one collection of object representations with another collection of object representations may include comparing at least a portion of one collection of object representations with at least a portion of the other

collection of object representations. In some embodiments, collections of object representations may be compared individually. In some aspects, comparing of individual collections of object representations may include comparing one or more object representations of one collection of object representations with one or more object representations of another collection of object representations. In other aspects, comparing of object representations may include comparing one or more object properties of one object representation with one or more object properties of another object representation. In some designs, one or more object properties in the same category (i.e. Category **635**, etc.) can be compared. Comparing may include any techniques for comparing text, numbers, and/or other data. In further aspects, some object representations, object properties, and/or other elements of a collection of object representations can be omitted from comparison depending on implementation. In other embodiments, collections of object representations may be compared collectively as part of streams of collections of object representations. Collective comparing of collections of object representations may include any features, functionalities, and embodiments of the aforementioned individual comparing of collections of object representations. In some aspects, collective comparing of collections of object representations may include comparing one or more collections of object representations of one stream of collections of object representations with one or more collections of object representations of another stream of collections of object representations. In some designs, one or more corresponding (i.e. similarly ordered, temporally related, etc.) collections of object representations from the compared streams of collections of object representations can be compared. In other designs, Dynamic Time Warping (DTW) and/or other techniques can be utilized for comparison and/or aligning temporal sequences (i.e. streams of collections of object representations, etc.) that may vary in time or speed. In further aspects, some collections of object representations can be omitted from comparison depending on implementation. Any combination of the aforementioned and/or other elements or techniques can be utilized in alternate embodiments of the comparing. Comparing may be omitted where anticipating of a device's operation in circumstances including objects with various properties is not implemented. Comparing comprises any action or operation by or for a Decision-making Unit **540**, Similarity Comparison **125**, and/or other disclosed elements.

At step **9135**, a determination is made that there is at least a partial match between the new collection of object representations and the first collection of object representations. In some embodiments, determining at least a partial match between individually compared collections of object representations includes determining that a similarity between one or more portions of one collection of object representations and one or more portions of another collection of object representations exceeds a similarity threshold. In other embodiments, determining at least a partial match between individually compared collections of object representations includes determining at least a partial match between one or more portions of one collection of object representations and one or more portions of another collection of object representations. In further embodiments, determining at least a partial match between individually compared collections of object representations includes determining substantial similarity between one or more portions of one collection of object representations and one or more portions of another collection of object representations. A portion of a collection of object representations may include an object representation, an object property, and/or other portion or element of the collection of object representations. In further embodiments, determining at least a partial match between individually compared collections of object representations includes determining that the number or percentage of matching or substantially matching object representations of the compared collections of object representations exceeds a threshold number (i.e. 1, 2, 4, 7, 18, etc.) or threshold percentage (i.e. 41%, 62%, 79%, 85%, 93%, etc.). In some aspects, type of object representations, importance of object representations, and/or other elements or techniques relating to object representations can be utilized for determining similarity using object representations. In further aspects, some of the object representations can be omitted in determining similarity using object representations depending on implementation. In further embodiments, determining a match or substantial match between compared object representations includes determining that the number or percentage of matching or substantially matching object properties of the compared object representations exceeds a threshold number (i.e. 1, 2, 3, 6, 11, etc.) or a threshold percentage (i.e. 55%, 61%, 78%, 82%, 99%, etc.). In some aspects, categories of object properties, importance of object properties, and/or other elements or techniques relating to object properties can be utilized for determining similarity using object properties. In further aspects, some of the object properties can be omitted in determining similarity using object properties depending on implementation. In some designs, substantial similarity of individually compared collections of object representations can be achieved when a similarity between one or more portions of one collection of object representations and one or more portions of another collection of object representations exceeds a similarity threshold. In other designs, substantial similarity of individually compared collections of object representations can be achieved when the number or percentage of matching or substantially matching object representations of the compared collections of object representations exceeds a threshold number (i.e. 1, 2, 4, 7, 18, etc.) or threshold percentage (i.e. 41%, 62%, 79%, 85%, 93%, etc.). In further aspects, substantial similarity of compared object representations can be achieved when the number or percentage of matching or substantially matching object properties of the compared object representations exceeds a threshold number (i.e. 1, 2, 3, 6, 11, etc.) or a threshold percentage (i.e. 55%, 61%, 78%, 82%, 99%, etc.). In some embodiments, determining at least a partial match between collectively compared collections of object representations (i.e. streams of collections of object representations, etc.) includes determining that the number or percentage of matching or substantially matching collections of object representations of the compared streams of collections of object representations exceeds a threshold number (i.e. 1, 2, 4, 9, 33, 138, etc.) or threshold percentage (i.e. 39%, 58%, 77%, 88%, 94%, etc.). In some aspects, importance of collections of object representations, order of collections of object representations, and/or other elements or techniques relating to collections of object representations can be utilized for determining similarity of collectively compared collections of object representations or streams of collections of object representations. In further aspects, some of the collections of object representations can be omitted in determining similarity of collectively compared collections of object representations or streams of collections of object representations depending on implementation. In some designs, a threshold for a number or percentage similarity can be used to determine a match or substantial

match between any of the aforementioned elements. Any text, number, and/or other data similarity determination techniques can be used in any of the aforementioned similarity determinations. A partial match of any of the compared elements may include a substantially or otherwise similar match, and vice versa. Therefore, these terms may be used interchangeably herein depending on context. Although, substantial similarity or substantial match is frequently used herein, it should be understood that any level of similarity, however high or low, may be utilized as defined by the rules (i.e. thresholds, etc.) for similarity. Any combination of the aforementioned and/or other elements or techniques can be utilized in alternate embodiments. Determining may be omitted where anticipating of a device's operation in circumstances including objects with various properties is not implemented. Determining comprises any action or operation by or for a Decision-making Unit **540**, Similarity Comparison **125**, and/or other disclosed elements.

At step **9140**, the first one or more instruction sets for operating the device correlated with the first collection of object representations are executed. Executing may be performed in response to the aforementioned determining. Executing may be caused by DCADO Unit **100**, Artificial Intelligence Unit **110**, Modification Interface **130**, and/or other disclosed elements. An instruction set may be executed by a processor (i.e. Processor **11**, etc.), application program (i.e. Application Program **18**, etc.), logic circuit (i.e. Logic Circuit **250**, etc.), and/or other processing element. An instruction set may be executed or acted upon by an actuator. In some aspects, instruction sets (i.e. the one or more instruction sets for operating the device correlated with the first collection of object representations, etc.) anticipated or determined to be used or executed in a device's autonomous operation may be referred to as anticipatory instruction sets, alternate instruction sets, and/or other suitable name or reference. Therefore, these terms can be used interchangeably herein depending on context. Executing may include executing one or more alternate instruction sets (i.e. anticipatory instruction sets, etc.) instead of or prior to an instruction set that would have been executed in a regular course of execution. In some embodiments, executing may include modifying a register or other element of a processor with one or more alternate instruction sets. Executing may also include redirecting a processor to one or more alternate instruction sets. In other embodiments, processor may be or comprises a logic circuit. Executing may further include modifying an element of a logic circuit with one or more alternate instruction sets, redirecting the logic circuit to one or more alternate instruction sets, replacing the inputs into the logic circuit with one or more alternate inputs or instruction sets, and/or replacing the outputs from the logic circuit with one or more alternate outputs or instruction sets. Executing may further include replacing the inputs into an actuator with one or more alternate inputs or instruction sets. In further embodiments, a processor may run an application including instruction sets for operating a device. In some aspects, executing includes executing one or more alternate instruction sets as part of the application. In other aspects, executing includes modifying the application. In further aspects, executing includes redirecting the application to one or more alternate instruction sets. In further aspects, executing includes modifying one or more instruction sets of the application. In further aspects, executing includes modifying the application's source code, bytecode, intermediate code, compiled code, interpreted code, translated code, runtime code, assembly code, machine code, or other code. In further aspects, executing includes modifying memory, processor register, storage, repository or other element where the application's instruction sets are stored or used. In further aspects, executing includes modifying instruction sets used for operating an object of the application. In further aspects, executing includes modifying an element of a processor, an element of a device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input used in running the application. In further aspects, executing includes modifying the application at source code write time, compile time, interpretation time, translation time, linking time, loading time, runtime, or other time. In further aspects, executing includes modifying one or more of the application's lines of code, statements, instructions, functions, routines, subroutines, basic blocks, or other code segments. In further aspects, executing includes a manual, automatic, dynamic, just in time (JIT), or other instrumentation of the application. In further aspects, executing includes utilizing one or more of a .NET tool, .NET application programming interface (API), Java tool, Java API, operating system tool, independent tool, or other tool for modifying the application. In further aspects, executing includes utilizing a dynamic, interpreted, scripting, or other programming language. In further aspects, executing includes utilizing dynamic code, dynamic class loading, or reflection. In further aspects, executing includes utilizing assembly language. In further aspects, executing includes utilizing metaprogramming, self-modifying code, or an application modification tool. In further aspects, executing includes utilizing just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further aspects, executing includes utilizing dynamic expression creation, dynamic expression execution, dynamic function creation, or dynamic function execution. In further aspects, executing includes adding or inserting additional code into the application's code. In further aspects, executing includes modifying, removing, rewriting, or overwriting the application's code. In further aspects, executing includes branching, redirecting, extending, or hot swapping the application's code. Branching or redirecting an application's code may include inserting a branch, jump, or other means for redirecting the application's execution. Executing comprises any action or operation by or for a Processor **11**, Application Program **18**, Logic Circuit **250**, Modification Interface **130**, and/or other disclosed elements.

At step **9145**, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations are performed by the device. The one or more operations may be performed in response to the aforementioned executing. In some aspects, an operation includes any operation that can be performed by/with/on a computing enabled device. In other aspects, an operation includes any operation that can be performed by/with/on an actuator. In further aspects, an operation includes any operation that can be performed by/with/on a computer. In general, an operation includes any operation that can be performed by/with/on a device or element thereof. One of ordinary skill in art will recognize that, while all possible variations of operations by/with/on a device are too voluminous to describe and limited only by the device's design and/or user's utilization, all operations are within the scope of this disclosure in various implementations.

Referring to FIG. **33**, the illustration shows an embodiment of a method **9200** for learning and/or using a device's circumstances for autonomous device operation. In some aspects, the method can be used on a computing enabled

device or system to enable learning of a device's operation in circumstances including objects with various properties and enable autonomous device operation in similar circumstances. Method **9200** may include any action or operation of any of the disclosed methods such as method **9100**, **9300**, **9400**, **9500**, **9600**, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method **9200**.

At step **9205**, a first collection of object representations is received. Step **9205** may include any action or operation described in Step **9105** of method **9100** as applicable.

At step **9210**, a first one or more instruction sets for operating a device are received. Step **9210** may include any action or operation described in Step **9110** of method **9100** as applicable.

At step **9215**, the first collection of object representations correlated with the first one or more instruction sets for operating the device are learned. Step **9215** may include any action or operation described in Step **9115** and/or Step **9120** of method **9100** as applicable.

At step **9220**, a new collection of object representations is received. Step **9220** may include any action or operation described in Step **9125** of method **9100** as applicable.

At step **9225**, the first one or more instruction sets for operating the device correlated with the first collection of object representations are anticipated based on at least a partial match between the new collection of object representations and the first collection of object representations. Step **9225** may include any action or operation described in Step **9130** and/or Step **9135** of method **9100** as applicable.

At step **9230**, the first one or more instruction sets for operating the device correlated with the first collection of object representations are executed. Step **9230** may include any action or operation described in Step **9140** of method **9100** as applicable.

At step **9235**, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations are performed by the device. Step **9235** may include any action or operation described in Step **9145** of method **9100** as applicable.

Referring to FIG. **34**, the illustration shows an embodiment of a method **9300** for learning and/or using a device's circumstances for autonomous device operation. In some aspects, the method can be used on a computing enabled device or system to enable learning of a device's operation in circumstances including objects with various properties and enable autonomous device operation in similar circumstances. Method **9300** may include any action or operation of any of the disclosed methods such as method **9100**, **9200**, **9400**, **9500**, **9600**, and/or others.

Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method **9300**.

At step **9305**, a first stream of collections of object representations is received. Step **9305** may include any action or operation described in Step **9105** of method **9100** as applicable.

At step **9310**, a first one or more instruction sets for operating a device are received. Step **9310** may include any action or operation described in Step **9110** of method **9100** as applicable.

At step **9315**, the first stream of collections of object representations is correlated with the first one or more instruction sets for operating the device. Step **9315** may include any action or operation described in Step **9115** of method **9100** as applicable.

At step **9320**, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device are stored. Step **9320** may include any action or operation described in Step **9120** of method **9100** as applicable.

At step **9325**, a new stream of collections of object representations is received. Step **9325** may include any action or operation described in Step **9125** of method **9100** as applicable.

At step **9330**, the new stream of collections of object representations is compared with the first stream of collections of object representations. Step **9330** may include any action or operation described in Step **9130** of method **9100** as applicable.

At step **9335**, a determination is made that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. Step **9335** may include any action or operation described in Step **9135** of method **9100** as applicable.

At step **9340**, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations are executed. Step **9340** may include any action or operation described in Step **9140** of method **9100** as applicable.

At step **9345**, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations are performed by the device. Step **9345** may include any action or operation described in Step **9145** of method **9100** as applicable.

Referring to FIG. **35**, the illustration shows an embodiment of a method **9400** for learning and/or using a device's circumstances for autonomous device operation. In some aspects, the method can be used on a computing enabled device or system to enable learning of a device's operation in circumstances including objects with various properties and enable autonomous device operation in similar circumstances. Method **9400** may include any action or operation of any of the disclosed methods such as method **9100**, **9200**, **9300**, **9500**, **9600**, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method **9400**.

At step **9405**, a first collection of object representations is received. Step **9405** may include any action or operation described in Step **9105** of method **9100** as applicable.

At step **9410**, a first one or more inputs are received, wherein the first one or more inputs are also received by a logic circuit, and wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device. Step **9410** may include any action or operation described in Step **9110** of method **9100** as applicable.

At step **9415**, the first collection of object representations is correlated with the first one or more inputs. Step **9415** may include any action or operation described in Step **9115** of method **9100** as applicable.

At step **9420**, the first collection of object representations correlated with the first one or more inputs are stored. Step **9420** may include any action or operation described in Step **9120** of method **9100** as applicable.

At step **9425**, a new collection of object representations is received. Step **9425** may include any action or operation described in Step **9125** of method **9100** as applicable.

At step **9430**, the new collection of object representations is compared with the first collection of object representations. Step **9430** may include any action or operation described in Step **9130** of method **9100** as applicable.

At step **9435**, a determination is made that there is at least a partial match between the new collection of object representations and the first collection of object representations. Step **9435** may include any action or operation described in Step **9135** of method **9100** as applicable.

At step **9440**, the first one or more inputs correlated with the first collection of object representations are received by the logic circuit. Step **9440** may include any action or operation described in Step **9140** of method **9100** as applicable.

At step **9445**, one or more operations defined by one or more outputs for operating the device produced by the logic circuit are performed by the device. Step **9445** may include any action or operation described in Step **9145** of method **9100** as applicable.

Referring to FIG. **36**, the illustration shows an embodiment of a method **9500** for learning and/or using a device's circumstances for autonomous device operation. In some aspects, the method can be used on a computing enabled device or system to enable learning of a device's operation in circumstances including objects with various properties and enable autonomous device operation in similar circumstances. Method **9500** may include any action or operation of any of the disclosed methods such as method **9100**, **9200**, **9300**, **9400**, **9600**, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method **9500**.

At step **9505**, a first collection of object representations is received. Step **9505** may include any action or operation described in Step **9105** of method **9100** as applicable.

At step **9510**, a first one or more outputs are received, the first one or more outputs transmitted from a logic circuit, wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device. Step **9510** may include any action or operation described in Step **9110** of method **9100** as applicable.

At step **9515**, the first collection of object representations is correlated with the first one or more outputs. Step **9515** may include any action or operation described in Step **9115** of method **9100** as applicable.

At step **9520**, the first collection of object representations correlated with the first one or more outputs are stored. Step **9520** may include any action or operation described in Step **9120** of method **9100** as applicable.

At step **9525**, a new collection of object representations is received. Step **9525** may include any action or operation described in Step **9125** of method **9100** as applicable.

At step **9530**, the new collection of object representations is compared with the first collection of object representations. Step **9530** may include any action or operation described in Step **9130** of method **9100** as applicable.

At step **9535**, a determination is made that there is at least a partial match between the new collection of object representations and the first collection of object representations. Step **9535** may include any action or operation described in Step **9135** of method **9100** as applicable.

At step **9540**, one or more operations defined by the first one or more outputs correlated with the first collection of object representations are performed by the device. Step **9540** may include any action or operation described in Step **9145** of method **9100** as applicable.

Referring to FIG. **37**, the illustration shows an embodiment of a method **9600** for learning and/or using a device's circumstances for autonomous device operation. In some aspects, the method can be used on a computing enabled device or system to enable learning of a device's operation in circumstances including objects with various properties and enable autonomous device operation in similar circumstances. Method **9600** may include any action or operation of any of the disclosed methods such as method **9100**, **9200**, **9300**, **9400**, **9500**, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method **9600**.

At step **9605**, a first collection of object representations is received. Step **9605** may include any action or operation described in Step **9105** of method **9100** as applicable.

At step **9610**, a first one or more inputs are received, wherein the first one or more inputs are also received by an actuator, and wherein the actuator is configured to receive inputs and perform motions. Step **9610** may include any action or operation described in Step **9110** of method **9100** as applicable.

At step **9615**, the first collection of object representations is correlated with the first one or more inputs. Step **9615** may include any action or operation described in Step **9115** of method **9100** as applicable.

At step **9620**, the first collection of object representations correlated with the first one or more inputs are stored. Step **9620** may include any action or operation described in Step **9120** of method **9100** as applicable.

At step **9625**, a new collection of object representations is received. Step **9625** may include any action or operation described in Step **9125** of method **9100** as applicable.

At step **9630**, the new collection of object representations is compared with the first collection of object representations. Step **9630** may include any action or operation described in Step **9130** of method **9100** as applicable.

At step **9635**, a determination is made that there is at least a partial match between the new collection of object representations and the first collection of object representations. Step **9635** may include any action or operation described in Step **9135** of method **9100** as applicable.

At step **9640**, the first one or more inputs correlated with the first collection of object representations are received by the actuator. Step **9640** may include any action or operation described in Step **9140** of method **9100** as applicable.

At step **9645**, one or more motions defined by the first one or more inputs correlated with the first collection of object representations are performed by the actuator. Step **9645** may include any action or operation described in Step **9145** of method **9100** as applicable.

Referring to FIG. **38**, in some exemplary embodiments, Device **98** may be or include Loader **98***a*. Loader **98***a* may be operated by User **50** in person or remotely. Loader **98***a* may include or be coupled to one or more Sensors **92** (i.e. collectively referred to as Sensor **92**, etc.) such as Camera **92***a*, Microphone **92***b*, Lidar **92***c*, Radar **92***d*, Sonar **92***e*, etc. and/or Object Processing Unit **93** that can detect Objects **615***aa*-**615***ad*, and/or other elements or information in Loader's **98***a* surrounding. Object Processing Unit **93** may include Picture Recognizer **94***a*, Sound Recognizer **94***b*,

Lidar Processing Unit **94***c*, Radar Processing Unit **94***d*, Sonar Processing Unit **94***e*, and/or other elements or functionalities as applicable. Object Processing Unit **93** may create or generate one or more (i.e. stream, etc.) Collections of Object Representations **525** comprising Object Representations **625**, Object Properties **630**, and/or other elements or information representing Objects **615** detected in Loader's **98***a* surrounding. Loader **98***a* may also include or be controlled by Logic Circuit **250** (i.e. microcontroller, etc.), Processor **11** (i.e. including any Application Program **18** running thereon, etc.), and/or other processing element that receives User's **50** (i.e. operator's, etc.) operating directions and causes desired operations with Loader **98***a* such as moving, maneuvering, collecting, lifting, unloading, and/or others. User **50** can interact with Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element through inputting operating directions via Human-machine Interface **23** such as one or more steering wheels, levers, pedals, buttons, or other input devices. For instance, responsive to User's **50** manipulating a steering wheel and one or more levers, Logic Circuit **250** or Processor **11** may cause Loader's **98***a* arm with bucket to collect a load, one or more motors or other actuators to move or maneuver Loader **98***a*, lifting system (i.e. hydraulic, pneumatic, mechanical, electrical, etc.) to lift a load, and/or arm with bucket to unload a load. Loader **98***a* may also include or be coupled to DCADO Unit **100**. DCADO Unit **100** may be embedded (i.e. integrated, etc.) into or coupled to Loader's **98***a* Logic Circuit **250**, Processor **11**, and/or other processing element. DCADO Unit **100** may also be a program embedded (i.e. integrated, etc.) into or interfaced with Application Program **18** running on Processor **11** and/or other processing element. DCADO Unit **100** can obtain Instruction Sets **526** used or executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element. In some aspects, Instruction Sets **526** may include one or more inputs into or outputs from Loader's **98***a* Logic Circuit **250** (i.e. microcontroller, etc.). In other aspects, Instruction Sets **526** may include one or more instruction sets from Loader's **98***a* Processor's **11** registers or other components. In further aspects, Instruction Sets **526** may include one or more instruction sets used or executed in Application Program **18**. DCADO Unit **100** may also optionally obtain any Extra Info **527** (i.e. time, location, computed, contextual, and/or other information, etc.) related to Loader's **98***a* operation. As User **50** operates Loader **98***a* in circumstances including objects with various properties as shown, DCADO Unit **100** may learn Loader's **98***a* operations in these circumstances by correlating Collections of Object Representations **525** representing Objects **615** detected in Loader's **98***a* surrounding with one or more Instruction Sets **526** used or executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element. Any Extra Info **527** related to Loader's **98***a* operation may also optionally be correlated with Collections of Object Representations **525**. DCADO Unit **100** may store this knowledge into Knowledgebase **530** (i.e. Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.). In the future, DCADO Unit **110** may compare incoming Collections of Object Representations **525** representing Objects **615** detected in Loader's **98***a* surrounding with previously learned Collections of Object Representations **525** including optionally using any Extra Info **527** for enhanced decision making. If substantially similar or at least a partial match is found or determined, the Instruction Sets **526** correlated with the previously learned Collections of Object Representations **525** can be autonomously executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element, thereby enabling autonomous operation of Loader **98***a* in similar circumstances as in previously learned ones. For instance, Loader **98***a* comprising DCADO Unit **100** may learn User **50**-directed collecting, moving, maneuvering, lifting, unloading, and/or other operations in a circumstance that includes Rock **615***aa*, Pile of Material **615***ab*, Person **615***ac*, Truck **615***ad*, and/or other Objects **615** among which Loader **98***a* may need to maneuver and/or with which Loader **98***a* may need to interact. In the future, when a circumstance that includes Objects **615** with similar Object Properties **630** is encountered, Loader **98***a* may implement collecting, moving, maneuvering, lifting, and/or unloading operations autonomously.

In some embodiments, DCADO Unit **100** may reside on Server **96** accessible over Network **95** as previously described. In such embodiments, any number of Loaders **98***a* may connect to such remote DCADO Unit **100** and the remote DCADO Unit **100** may learn their operations in circumstances including objects with various properties. In turn, any number of Loaders **98***a* can utilize the remote DCADO Unit **100** for autonomous operation in circumstances including objects with various properties. For example, multiple operators (i.e. Users **50**, etc.) may operate their Loaders **98***a* that are configured to transmit their operations in circumstances including objects with various properties to a remote DCADO Unit **100**. Such remote DCADO Unit **100** enables learning of the operators' collective knowledge of operating Loaders **98** in circumstances including objects with various properties. Any number of Loaders **98** can utilize such collective knowledge comprised in the remote DCADO Unit **100** for their autonomous operation. Any of the disclosed elements such as Artificial Intelligence Unit **110**, Knowledgebase **530**, and/or others may reside on Server **96**, and any combination of local and remote elements can be implemented in alternate embodiments.

In some embodiments, Loader **98***a* may include or be coupled to a plurality of Sensors **92** and/or their corresponding Object Processing Units **93**. In one example, multiple Sensors **92** may detect objects and/or their properties from different angles or on different sides of Loader **98***a*. In another example, one or more Sensors **92** may be placed on different sub-devices, sub-systems, or elements of Loader **98***a*. For instance, one Sensor **92** may be placed on the roof of Loader **98***a*, another Sensor **92** may be placed on the arm of Loader **98***a*, and an additional Sensor **92** may be placed on the bucket of Loader **98***a*. In some designs where multiple Sensors **92** are placed on different sub-devices, sub-systems, or elements of Loader **98***a*, multiple DCADO Units **100** can be utilized (i.e. one DCADO Unit **100** for each Sensor **92** or group of Sensors **92** and/or their corresponding Object Processing Units **93**, etc.). In such designs, as User **50** operates Loader **98***a* in circumstances including objects with various properties, a particular DCADO Unit **100** may learn operations of Loader's **98***a* sub-device, sub-system, or element in these circumstances by correlating Collections of Object Representations **525** representing Objects **615** detected by Sensor **92** on the sub-device, sub-system, or element assigned to the DCADO Unit **100** with one or more Instruction Sets **526** used or executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element. The learning and/or decision making in Loader's **98***a* operation can, therefore, be performed per individual sub-device, sub-system, or element. In other designs where multiple Sensors **92** are placed

on different sub-devices, sub-systems, or elements of Loader **98***a*, as User **50** operates Loader **98***a* in circumstances including objects with various properties, a single DCADO Unit **100** may learn Loader's **98***a* operations in these circumstances by correlating collective Collections of Object Representations **525** representing Objects **615** detected by Sensors **92** on the sub-devices, sub-systems, or elements with one or more Instruction Sets **526** used or executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element.

In some embodiments, Loader **98***a* may include a plurality of Logic Circuits **250**, Processors **11**, Application Programs **18**, and/or other processing elements. In some aspects, one or more sub-devices, sub-systems, or elements of Loader **98***a* may be controlled by different processing elements. For example, one Processor **11** (i.e. including any Application Programs **18** running thereon, etc.) may control the moving system (i.e. drivetrain, powertrain, etc.) of Loader **98***a*, one Logic Circuit **250** may control an arm of Loader **98***a*, and an additional Logic Circuit **250** may control a bucket of Loader **98***a*. In some designs where multiple processing elements are utilized, multiple DCADO Units **100** can also be utilized (i.e. one DCADO Unit **100** for each processing element, etc.). In such designs, as User **50** operates Loader **98***a* in circumstances including objects with various properties, a particular DCADO Unit **100** may learn Loader's **98***a* operations in these circumstances by correlating Collections of Object Representations **525** with one or more Instruction Sets **526** used or executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element assigned to the DCADO Unit **100**. The learning and/or decision making in Loader's **98***a* operation can, therefore, be performed per individual processing element. In other designs where multiple processing elements are utilized, as User **50** operates Loader **98***a* in circumstances including objects with various properties, a single DCADO Unit **100** may learn Loader's **98***a* operations in these circumstances by correlating Collections of Object Representations **525** with collective Instruction Sets **526** used or executed by a plurality of Logic Circuits **250**, Processors **11**, Application Programs **18**, and/or other processing elements.

In some embodiments, a combination of DCADO Unit **100** and other systems and/or techniques can be utilized to implement Loader's **98***a* operation. In one example, DCADO Unit **100** may be a primary or preferred system for implementing Loader's **98***a* operation. While operating autonomously under the control of DCADO Unit **100**, Loader **98***a* may encounter a circumstance including objects with various properties that has not been encountered or learned before. In such situations, User **50** and/or non-DCADO system may take control of Loader's **98***a* operation. DCADO Unit **100** may take control again when Loader **98***a* encounters a previously learned circumstance including objects with various properties. Naturally, DCADO Unit **100** can learn Loader's **98***a* operation in the circumstances while User **50** and/or non-DCADO system is in control of Loader **98***a*, thereby reducing or eliminating the need for future involvement of User **50** and/or non-DCADO system. For instance, one User **50** can control or assist in controlling multiple Loaders **98***a* comprising DCADO Units **100**. In such instances, User **50** can control or assist in controlling a Loader **98***a* that may encounter a circumstance including objects with various properties that has not been encountered or learned before while the Loaders **98***a* operating in previously learned circumstances can operate autonomously. In another example, User **50** and/or non-DCADO system may be a primary or preferred system for implementing Loader's **98***a* operation. While operating under the control of User **50** and/or non-DCADO system, User **50** and/or non-DCADO system may release control to DCADO Unit **100** for any reason (i.e. User **50** gets tired or distracted, non-DCADO system gets stuck or cannot make a decision, etc.), at which point Loader **98***a* can be controlled by DCADO Unit **100**. In some designs, DCADO Unit **100** may take control in certain special circumstances including objects with various properties where DCADO Unit **100** may offer superior performance even though User **50** and/or non-DCADO system may generally be preferred. Once Loader **98***a* leaves such special circumstances, DCADO Unit **100** may release control to User **50** and/or non-DCADO system. In general, DCADO Unit **100** can take control from, share control with, or release control to User **50**, non-DCADO system, and/or other system or process at any time, in any circumstances, and remain in control for any period of time as needed.

In some embodiments, DCADO Unit **100** may control one or more sub-devices, sub-systems, or elements of Loader **98***a* while User **50** and/or non-DCADO system may control other one or more sub-devices, sub-systems, or elements of Loader **98***a*. For example, User **50** and/or non-DCADO system may control the moving system (i.e. drivetrain, powertrain, etc.) of Loader **98***a*, while DCADO Unit **100** may control an arm and bucket of Loader **98***a*. Any other combination of controlling various sub-devices, sub-systems, or elements of Loader **98***a* by DCADO Unit **100** and User **50** and/or non-DCADO system can be implemented.

Referring to FIG. **39**, in some exemplary embodiments, Device **98** may be or include Boat **98***b*. Boat **98***b* may be operated by User **50** in person or remotely. Boat **98***b* may include or be coupled to one or more Sensors **92** (i.e. collectively referred to as Sensor **92**, etc.) such as Camera **92***a*, Microphone **92***b*, Lidar **92***c*, Radar **92***d*, Sonar **92***e*, etc. and/or Object Processing Unit **93** that can detect Objects **615***ba*-**615***bd*, and/or other elements or information in Boat's **98***b* surrounding. Object Processing Unit **93** may include Picture Recognizer **94***a*, Sound Recognizer **94***b*, Lidar Processing Unit **94***c*, Radar Processing Unit **94***d*, Sonar Processing Unit **94***e*, and/or other elements or functionalities as applicable. Object Processing Unit **93** may create or generate one or more (i.e. stream, etc.) Collections of Object Representations **525** comprising Object Representations **625**, Object Properties **630**, and/or other elements or information representing Objects **615** detected in Boat's **98***b* surrounding. Boat **98***b* may also include or be controlled by Logic Circuit **250** (i.e. microcontroller, etc.), Processor **11** (i.e. including any Application Program **18** running thereon, etc.), and/or other processing element that receives User's **50** (i.e. operator's, etc.) operating directions and causes desired operations with Boat **98***b* such as moving, maneuvering, and/or other operations. User **50** can interact with Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element through inputting operating directions via Human-machine Interface **23** such as one or more steering wheels, levers, pedals, buttons, or other input devices. For instance, responsive to User's **50** manipulating a steering wheel and one or more levers, Logic Circuit **250** or Processor **11** may cause one or more motors or other actuators to move or maneuver Boat **98***b*. Boat **98***b* may also include or be coupled to DCADO Unit **100**. DCADO Unit **100** may be embedded (i.e. integrated, etc.) into or coupled to Boat's **98***b* Logic Circuit **250**, Processor **11**, and/or other processing element. DCADO Unit **100** may also be a

program embedded (i.e. integrated, etc.) into or interfaced with Application Program **18** running on Processor **11** and/or other processing element. DCADO Unit **100** can obtain Instruction Sets **526** used or executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element. In some aspects, Instruction Sets **526** may include one or more inputs into or outputs from Boat's **98***b* Logic Circuit **250** (i.e. microcontroller, etc.). In other aspects, Instruction Sets **526** may include one or more instruction sets from Boat's **98***b* Processor's **11** registers or other components. In further aspects, Instruction Sets **526** may include one or more instruction sets used or executed in Application Program **18**. DCADO Unit **100** may also optionally obtain any Extra Info **527** (i.e. time, location, computed, contextual, and/or other information, etc.) related to Boat's **98***b* operation. As User **50** operates Boat **98***b* in circumstances including objects with various properties as shown, DCADO Unit **100** may learn Boat's **98***b* operations in these circumstances by correlating Collections of Object Representations **525** representing Objects **615** detected in Boat's **98***b* surrounding with one or more Instruction Sets **526** used or executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element. Any Extra Info **527** related to Boat's **98***b* operation may also optionally be correlated with Collections of Object Representations **525**. DCADO Unit **100** may store this knowledge into Knowledgebase **530** (i.e. Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.). In the future, DCADO Unit **110** may compare incoming Collections of Object Representations **525** representing Objects **615** detected in Boat's **98***b* surrounding with previously learned Collections of Object Representations **525** including optionally using any Extra Info **527** for enhanced decision making. If substantially similar or at least a partial match is found or determined, the Instruction Sets **526** correlated with the previously learned Collections of Object Representations **525** can be autonomously executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element, thereby enabling autonomous operation of Boat **98***b* in similar circumstances as in previously learned ones. For instance, Boat **98***b* comprising DCADO Unit **100** may learn User **50**-directed moving, maneuvering, and/or other operations in a circumstance that includes Fishing Boat **615***ba*, Lighthouse **615***bb*, Sailboat **615***bc*, Cruise Ship **615***bd*, and/or other Objects **615** among which Boat **98***b* may need to maneuver. In the future, when a circumstance that includes Objects **615** with similar Object Properties **630** is encountered, Boat **98***b* may implement moving, maneuvering, and/or other operations autonomously. In some aspects, the shore (not enumerated) or any part thereof (i.e. cliff, ridge, beach, etc.) may be detected as an Object **615** itself, which may then be learned and used in autonomous operation of Boat **98***b*.

Referring to FIG. **40**, in some exemplary embodiments, an Area of Interest **450** can be utilized. In one example, Area of Interest **450** may include a radial, circular, elliptical, or other such area around Boat **98***b*. In another example, Area of Interest **450** may include a triangular, rectangular, octagonal, or other such area around Boat **98***b*. In a further example, Area of Interest **450** may include a spherical, cubical, pyramid-like, or other such area around Boat **98***b* as applicable to 3D space. Any other Area of Interest **450** shape can be utilized depending on implementation. The shape and/or size of Area of Interest **450** can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. Utilizing Area of Interest **450** enables DCADO Unit **100** to focus on Boats **98***b* immediate surrounding, thereby avoiding extraneous detail in the rest of the surrounding. In some aspects, Area of Interest **450** can be subdivided into sub-areas (i.e. sub-circles, sub-rectangles, sub-spheres, etc.). Sub-areas can be used to classify the surrounding by distance from Boat **98***b*. For example, the surrounding closer to Boat **98***b* may be more important and may be assigned higher importance index or weight. As User **50** operates Boat **98***b* in circumstances including objects with various properties as shown, DCADO Unit **100** may learn Boat's **98***b* operations in these circumstances by correlating Collections of Object Representations **525** representing Objects **615** detected in Area of Interest **450** around Boat **98***b* with one or more Instruction Sets **526** used or executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element. Any Extra Info **527** related to Boat's **98***b* operation may also optionally be correlated with Collections of Object Representations **525**. DCADO Unit **100** may store this knowledge into Knowledgebase **530** (i.e. Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.). In the future, DCADO Unit **110** may compare incoming Collections of Object Representations **525** representing Objects **615** detected in Area of Interest **450** around Boat **98***b* with previously learned Collections of Object Representations **525** including optionally using any Extra Info **527** for enhanced decision making. If substantially similar or at least a partial match is found or determined, the Instruction Sets **526** correlated with the previously learned Collections of Object Representations **525** can be autonomously executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element, thereby enabling autonomous operation of Boat **98***b* in similar Areas of Interest **450** as in previously learned ones. For instance, Boat **98***b* comprising DCADO Unit **100** may learn User **50**-directed moving, maneuvering, and/or other operations in an Area of Interest **450** that includes Fishing Boat **615***ba*, Lighthouse **615***bb*, Cruise Ship **615***bd*, and/or other Objects **615** among which Boat **98***b* may need to maneuver. In the future, when an Area of Interest **450** that includes Objects **615** with similar Object Properties **630** is encountered, Boat **98***b* may implement moving, maneuvering, and/or other operations autonomously.

The features, functionalities, and embodiments described with respect to Loader **98***a* and Boat **98***b* can be implemented in any situation where Device **98** may need to autonomously maneuver among, interact with, or perform other operations relative to objects in its surrounding. Therefore, the features, functionalities, and embodiments described with respect to Loader **98***a* and Boat **98***b* can similarly be implemented on any computing enabled machine such as a bulldozer, an excavator, a crane, a forklift, a truck, a construction machine, an assembly machine, an object handling machine, an object dispensing machine, a sorting machine, a restocking machine, an industrial machine, an agricultural machine, a harvesting machine, a building control system, a home or other appliance, a toy, a robot, a tank, an aircraft, a vessel, a submarine, a ground vehicle, an aerial vehicle, an aquatic vehicle, and/or other computing-enabled machine or system.

In yet some exemplary embodiments, Device **98** may be or include a control device such as a thermostat, control panel, remote or other controller, and/or other control device. For instance, a thermostat comprising DCADO Unit **100** may learn User's **50** setting temperature of an air

conditioning system controlled by the thermostat in a circumstance that includes User **50** and/or other persons entering or being present in a room. In the future, when a circumstance that includes User **50** and/or other persons entering or being present in the room is encountered, thermostat may implement setting temperature of the air conditioning system autonomously. In some aspects, a control device may be included in the device being controlled (i.e. control panel of an oven, refrigerator, fixture, etc.). In other aspects, a control device may be separate from the device being controlled (i.e. remote controller of a television device, etc.). In yet further exemplary embodiments, Device **98** may be or include a mobile computer such as a smartphone, tablet, and/or other mobile computer. For instance, a smartphone comprising DCADO Unit **100** may learn User **50**-directed playing a music file, setting a vibrate mode, and/or other operations in a circumstance that includes objects with various properties. In the future, when a circumstance that includes objects with similar properties is encountered, smartphone may implement playing music file, setting vibrate mode, and/or other operations autonomously. In general, Device **98** may be or include any movable, stationary, or other device. One of ordinary skill in art will understand that Device **98** may be or include any device that can implement and/or benefit from the functionalities described herein.

It must be noted that as used herein and in the appended claims, the singular forms "a", "an", and "the" include plural referents unless the context clearly dictates otherwise.

A number of embodiments have been described herein. While this disclosure contains many specific implementation details, these should not be construed as limitations on the scope of the disclosure or of what may be claimed, but rather as descriptions of features specific to particular embodiments. It should be understood that various modifications can be made without departing from the spirit and scope of the disclosure. The logic flows depicted in the figures do not require the particular order shown, or sequential order, to achieve desirable results. In addition, other or additional steps, elements, or connections can be included, or some of the steps, elements, or connections can be eliminated, or a combination thereof can be utilized in the described flows, illustrations, or descriptions. Further, the various aspects of the disclosed devices, apparatuses, systems, and/or methods can be combined in whole or in part with each other to produce additional implementations. Moreover, separation of various components in the embodiments described herein should not be understood as requiring such separation in all embodiments, and it should be understood that the described components can generally be integrated together in a single product or packaged into multiple products. Accordingly, other embodiments are within the scope of the following claims.

The invention claimed is:

1. A system comprising:

one or more non-transitory machine readable media storing machine readable code that, when executed, causes at least:

accessing a knowledgebase that includes one or more inputs for inputting at least a portion of a circumstance representation, wherein the one or more inputs are correlated with one or more instruction sets for operating a first device;

generating or receiving a circumstance representation, wherein the generated or the received circumstance representation represents a circumstance detected at least in part by: one or more sensors of the first device, or one or more sensors of a second device;

determining the one or more instruction sets for operating the first device at least by: inputting at least a portion of the generated or the received circumstance representation into the one or more inputs, and using a correlation between the one or more inputs and the one or more instruction sets for operating the first device; and

at least in response to the determining, executing the one or more instruction sets for operating the first device, wherein the first device or the second device autonomously performs one or more operations defined by the one or more instruction sets for operating the first device.

2. The system of claim **1**, wherein the one or more inputs for inputting the at least the portion of the circumstance representation include one or more inputs for inputting at least a portion of one or more object representations, and wherein the generated or the received circumstance representation includes one or more object representations.

3. The system of claim **2**, wherein at least an information related to the one or more instruction sets for operating the first device is learned in a learning process that includes operating the first device at least partially by a user.

4. The system of claim **3**, wherein the information related to the one or more instruction sets for operating the first device includes a weight.

5. The system of claim **2**, wherein: the correlation between the one or more inputs and the one or more instruction sets for operating the first device is learned in a learning process that includes operating the first device at least partially by a user, or an information related to the correlation between the one or more inputs and the one or more instruction sets for operating the first device is learned in a learning process that includes operating the first device at least partially by a user.

6. The system of claim **5**, wherein the information related to the correlation between the one or more inputs and the one or more instruction sets for operating the first device includes a weight.

7. The system of claim **2**, wherein at least a portion of the one or more instruction sets for operating the first device is learned in a learning process that includes operating the first device at least partially by a user.

8. The system of claim **2**, wherein the generated or the received circumstance representation represents the circumstance detected at least in part by the one or more sensors of the first device, and wherein the first device autonomously performs the one or more operations defined by the one or more instruction sets for operating the first device.

9. The system of claim **8**, wherein the one or more instruction sets for operating the first device are applied to the first device.

10. The system of claim **2**, wherein the generated or the received circumstance representation represents the circumstance detected at least in part by the one or more sensors of the second device, and wherein the second device autonomously performs the one or more operations defined by the one or more instruction sets for operating the first device.

11. The system of claim **10**, wherein the one or more instruction sets for operating the first device are applied to the second device.

12. The system of claim **2**, wherein the one or more inputs are further correlated with another one or more instruction sets for operating the first device, and wherein at least a portion of the one or more instruction sets for operating the

first device or at least an information related to the one or more instruction sets for operating the first device is learned in a learning process that includes operating the first device at least partially by a user, and wherein at least a portion of the another one or more instruction sets for operating the first device or at least an information related to the another one or more instruction sets for operating the first device is learned in another learning process that includes operating the first device at least partially by the user.

13. The system of claim **2**, wherein the one or more inputs are further correlated with another one or more instruction sets for operating the first device, and wherein at least a portion of the one or more instruction sets for operating the first device or at least an information related to the one or more instruction sets for operating the first device is learned in a learning process that includes operating the first device at least partially by a user, and wherein at least a portion of the another one or more instruction sets for operating the first device or at least an information related to the another one or more instruction sets for operating the first device is learned in another learning process that includes operating the first device at least partially by another user.

14. The system of claim **2**, wherein the one or more inputs are further correlated with another one or more instruction sets for operating a third device, and wherein at least a portion of the one or more instruction sets for operating the first device or at least an information related to the one or more instruction sets for operating the first device is learned in a learning process that includes operating the first device at least partially by a user, and wherein at least a portion of the another one or more instruction sets for operating the third device or at least an information related to the another one or more instruction sets for operating the third device is learned in another learning process that includes operating the third device at least partially by the user.

15. The system of claim **2**, wherein the one or more inputs are further correlated with another one or more instruction sets for operating a third device, and wherein at least a portion of the one or more instruction sets for operating the first device or at least an information related to the one or more instruction sets for operating the first device is learned in a learning process that includes operating the first device at least partially by a user, and wherein at least a portion of the another one or more instruction sets for operating the third device or at least an information related to the another one or more instruction sets for operating the third device is learned in another learning process that includes operating the third device at least partially by another user.

16. The system of claim **2**, wherein the one or more inputs are further correlated with another one or more instruction sets for operating a third device.

17. The system of claim **2**, wherein the one or more inputs are further correlated with another one or more instruction sets for operating a third device, and wherein the machine readable code, when executed, further causes at least:

generating or receiving another circumstance representation, wherein the generated or the received another circumstance representation represents a circumstance detected at least in part by one or more sensors of a fourth device;

determining the another one or more instruction sets for operating the third device at least by: inputting at least a portion of the generated or the received another circumstance representation into the one or more inputs, and using a correlation between the one or more inputs and the another one or more instruction sets for operating the third device; and

at least in response to the determining the another one or more instruction sets for operating the third device, executing the another one or more instruction sets for operating the third device, wherein the fourth device autonomously performs one or more operations defined by the another one or more instruction sets for operating the third device.

18. The system of claim **2**, wherein the one or more object representations of the generated or the received circumstance representation represent one or more objects in: the first device's surrounding, or the second device's surrounding, and wherein the first device's surrounding includes at least one of:

an area of interest around the first device, or

an area within a threshold distance from the first device, and wherein the second device's surrounding includes at least one of:

an area of interest around the second device, or

an area within a threshold distance from the second device.

19. The system of claim **2**, wherein the one or more object representations of the generated or the received circumstance representation include one or more information about one or more objects.

20. The system of claim **2**, wherein the one or more object representations of the generated or the received circumstance representation include one or more coordinates of one or more objects.

21. The system of claim **2**, wherein the one or more object representations of the generated or the received circumstance representation include at least one of: one or more information about one or more distances of one or more objects relative to the first device, one or more information about one or more angles of one or more objects relative to the first device, one or more information about one or more distances of one or more objects relative to the second device, or one or more information about one or more angles of one or more objects relative to the second device.

22. The system of claim **2**, wherein the one or more object representations of the generated or the received circumstance representation include one or more three dimensional representations of one or more objects.

23. The system of claim **2**, wherein the one or more object representations of the generated or the received circumstance representation include: one or more digital pictures that depict one or more objects, or one or more digital pictures that depict one or more representations of one or more objects.

24. The system of claim **2**, wherein the machine readable code, when executed, further causes at least:

modifying the one or more instruction sets for operating the first device,

wherein the determining the one or more instruction sets for operating the first device at least by: the inputting the at least the portion of the generated or the received circumstance representation into the one or more inputs, and the using the correlation between the one or more inputs and the one or more instruction sets for operating the first device includes determining the modified one or more instruction sets for operating the first device at least by: the inputting the at least the portion of the generated or the received circumstance representation into the one or more inputs, and using the correlation between the one or more inputs and the modified one or more instruction sets for operating the first device, and

wherein the executing the one or more instruction sets for operating the first device includes executing the modified one or more instruction sets for operating the first device, and

wherein the first device or the second device autonomously performing the one or more operations defined by the one or more instruction sets for operating the first device includes the first device or the second device autonomously performing one or more operations defined by the modified one or more instruction sets for operating the first device.

25. The system of claim **2**, wherein the machine readable code, when executed, further causes at least:

modifying: the one or more instruction sets for operating the first device, or a copy of the one or more instruction sets for operating the first device, and

wherein the executing the one or more instruction sets for operating the first device includes: executing the modified one or more instruction sets for operating the first device, or executing the modified the copy of the one or more instruction sets for operating the first device, and

wherein the first device or the second device autonomously performing the one or more operations defined by the one or more instruction sets for operating the first device includes: the first device or the second device autonomously performing one or more operations defined by the modified one or more instruction sets for operating the first device, or the first device or the second device autonomously performing one or more operations defined by the modified the copy of the one or more instruction sets for operating the first device.

26. The system of claim **2**, wherein the machine readable code, when executed, further causes at least:

modifying: the generated or the received circumstance representation, or a copy of the generated or the received circumstance representation, and

wherein the inputting the at least the portion of the generated or the received circumstance representation into the one or more inputs includes: inputting at least a portion of the modified the generated or the received circumstance representation into the one or more inputs, or inputting at least a portion of the modified the copy of the generated or the received circumstance representation into the one or more inputs.

27. The system of claim **2**, wherein the system further comprising:

a server that receives from the first device the one or more instruction sets for operating the first device, and wherein the second device receives from the server the one or more instruction sets for operating the first device, and wherein the second device autonomously performs the one or more operations defined by the first one or more instruction sets for operating the first device.

28. The system of claim **2**, wherein the knowledgebase is a neural network.

29. The system of claim **2**, wherein the knowledgebase includes an artificial intelligence system.

30. The system of claim **2**, wherein the one or more inputs are correlated with the one or more instruction sets for operating the first device using at least one or more connections, and wherein the using the correlation between the one or more inputs and the one or more instruction sets for operating the first device includes using at least one connection of the one or more connections between the one or more inputs and the one or more instruction sets for operating the first device.

31. The system of claim **2**, wherein the one or more inputs are one or more input nodes, and wherein the knowledgebase further includes one or more output nodes that include the one or more instruction sets for operating the first device, and wherein the one or more input nodes are connected with the one or more output nodes using at least one or more connections.

32. The system of claim **2**, wherein the knowledgebase further includes one or more outputs that include the one or more instruction sets for operating the first device, and wherein the correlation between the one or more inputs and the one or more instruction sets for operating the first device or an information related to the correlation between the one or more inputs and the one or more instruction sets for operating the first device is learned in a learning process that includes:

generating or receiving another circumstance representation, wherein the generated or the received another circumstance representation represents another circumstance detected at least in part by the one or more sensors of the first device;

obtaining or receiving the one or more instruction sets for operating the first device;

inputting at least a portion of the generated or the received another circumstance representation into the one or more inputs; and

applying the one or more instruction sets for operating the first device to the one or more outputs.

33. The system of claim **32**, wherein the learning process further includes:

back-propagating information from the one or more outputs to the one or more inputs.

34. The system of claim **2**, wherein the knowledgebase further includes one or more outputs that include the one or more instruction sets for operating the first device, and wherein the determining the one or more instruction sets for operating the first device is further performed at least by:

receiving the one or more instruction sets for operating the first device from the one or more outputs.

35. The system of claim **2**, wherein the generated or the received circumstance representation includes: a representation of the first device, or a representation of the second device.

36. The system of claim **2**, wherein the one or more instruction sets for operating the first device include one or more information about one or more states of: the first device, or a portion of the first device.

37. The system of claim **2**, wherein the one or more instruction sets for operating the first device include one or more representations of another one or more instruction sets for operating the first device.

38. The system of claim **2**, wherein the one or more instruction sets for operating the first device are further for operating at least a third device.

39. The system of claim **2**, wherein at least a portion of the received circumstance representation is received from an application for generating circumstance representations.

40. The system of claim **2**, wherein the generated or the received circumstance representation represents the circumstance detected at least in part by: the one or more sensors of the first device, or the one or more sensors of the second device at a time.

41. The system of claim **2**, wherein the generated or the received circumstance representation represents the circum-

stance detected at least in part by: the one or more sensors of the first device, or the one or more sensors of the second device during a time period.

42. The system of claim **2**, wherein the generated or the received circumstance representation is a data structure that includes one or more data about the circumstance detected at least in part by: the one or more sensors of the first device, or the one or more sensors of the second device, and wherein the circumstance detected at least in part by: the one or more sensors of the first device, or the one or more sensors of the second device includes one or more objects detected at least in part by: the one or more sensors of the first device, or the one or more sensors of the second device at a first time or during a first time period.

43. The system of claim **2**, wherein at least some parts of the system are included in: a single device, or multiple devices, and wherein the one or more sensors of the first device include at least one of: one or more cameras, one or more microphones, one or more radars, one or more lidars, one or more sonars, one or more apparatuses for detecting circumstances, or one or more apparatuses for detecting objects or object properties, and wherein the one or more sensors of the second device include at least one of: one or more cameras, one or more microphones, one or more radars, one or more lidars, one or more sonars, one or more apparatuses for detecting circumstances, or one or more apparatuses for detecting objects or object properties, and wherein an instruction set of the one or more instruction sets for operating the first device includes at least one of: only one instruction, multiple instructions, one or more commands, one or more keywords, one or more symbols, one or more operators, one or more variables, one or more parameters, one or more characters, one or more numbers, one or more values, one or more signals, one or more binary bits, one or more functions, one or more function references, one or more objects, one or more object references, one or more data structures, one or more data structure references, one or more states, one or more representations of one or more states, one or more inputs, one or more representations of one or more inputs, one or more codes, one or more data, or one or more information, and wherein the one or more object representations include: one object representation, multiple object representations, a collection of object representations, or a stream of collections of object representations, and wherein the one or more object representations include: one or more three dimensional representations of one or more objects, one or more digital pictures that depict one or more objects, one or more digital pictures that depict one or more representations of one or more objects, one or more information about one or more properties of one or more objects, or one or more computer representations of one or more objects, and wherein the at least the portion of the generated or the received circumstance representation includes: one portion of the generated or the received circumstance representation, multiple portions of the generated or the received circumstance representation, or the entire generated or received circumstance representation, and wherein the at least the portion of the generated or the received circumstance representation includes: one portion of an object representation of the one or more object representations of the generated or the received circumstance representation, multiple portions of an object representation of the one or more object representations of the generated or the received circumstance representation, multiple portions of multiple object representations of the one or more object representations of the generated or the received circumstance representation, one object representation of the one or more object representations of the generated or the received circumstance representation, multiple object representations of the one or more object representations of the generated or the received circumstance representation, or the entire one or more object representations of the generated or the received circumstance representation.

44. The system of claim **2**, wherein at least a portion of the knowledgebase is stored in or on at least one of: at least one non-transitory machine readable medium of the one or more non-transitory machine readable media, another one or more non-transitory machine readable media, one or more volatile memories, one or more non-volatile memories, one or more storage devices, or one or more storage systems, and wherein the system further comprises:

one or more processors, wherein the machine readable code is executed by the one or more processors, and wherein the one or more processors cause the accessing, the generating or the receiving, the determining, and the executing.

45. A first device comprising:

a knowledgebase that includes one or more inputs for inputting at least a portion of a circumstance representation, wherein the one or more inputs are correlated with one or more instruction sets for operating a second device;

one or more sensors;

one or more processors; and

one or more non-transitory machine readable media storing machine readable code that, when executed by the one or more processors, causes the one or more processors to perform at least:

generating a circumstance representation, wherein the generated circumstance representation represents a circumstance detected at least in part by the one or more sensors;

determining the one or more instruction sets for operating the second device at least by: inputting at least a portion of the generated circumstance representation into the one or more inputs, and using a correlation between the one or more inputs and the one or more instruction sets for operating the second device; and

at least in response to the determining, executing the one or more instruction sets for operating the second device, wherein the first device autonomously performs one or more operations defined by the one or more instruction sets for operating the second device.

46. The first device of claim **45**, wherein the one or more inputs for inputting the at least the portion of the circumstance representation include one or more inputs for inputting at least a portion of one or more object representations, and wherein the generated circumstance representation includes one or more object representations, and wherein the first device is a first vehicle, and wherein the second device is a second vehicle.

47. The first device of claim **46**, wherein at least an information related to the one or more instruction sets for operating the second device is learned in a learning process that includes operating the second device at least partially by a user.

48. The first device of claim **47**, wherein the information related to the one or more instruction sets for operating the second device includes a weight.

49. The first device of claim **46**, wherein: the correlation between the one or more inputs and the one or more instruction sets for operating the second device is learned in a learning process that includes operating the second device at least partially by a user, or an information related to the

correlation between the one or more inputs and the one or more instruction sets for operating the second device is learned in a learning process that includes operating the second device at least partially by a user.

50. The first device of claim **49**, wherein the information related to the correlation between the one or more inputs and the one or more instruction sets for operating the second device includes a weight.

51. The first device of claim **46**, wherein at least a portion of the one or more instruction sets for operating the second device is learned in a learning process that includes operating the second device at least partially by a user.

52. The first device of claim **46**, wherein the one or more instruction sets for operating the second device are applied to the first device.

53. The first device of claim **46**, wherein the one or more inputs are further correlated with another one or more instruction sets for operating the second device, and wherein at least a portion of the one or more instruction sets for operating the second device or at least an information related to the one or more instruction sets for operating the second device is learned in a learning process that includes operating the second device at least partially by a user, and wherein at least a portion of the another one or more instruction sets for operating the second device or at least an information related to the another one or more instruction sets for operating the second device is learned in another learning process that includes operating the second device at least partially by the user.

54. The first device of claim **46**, wherein the one or more inputs are further correlated with another one or more instruction sets for operating the second device, and wherein at least a portion of the one or more instruction sets for operating the second device or at least an information related to the one or more instruction sets for operating the second device is learned in a learning process that includes operating the second device at least partially by a user, and wherein at least a portion of the another one or more instruction sets for operating the second device or at least an information related to the another one or more instruction sets for operating the second device is learned in another learning process that includes operating the second device at least partially by another user.

55. The first device of claim **46**, wherein the one or more inputs are further correlated with another one or more instruction sets for operating a third device, and wherein at least a portion of the one or more instruction sets for operating the second device or at least an information related to the one or more instruction sets for operating the second device is learned in a learning process that includes operating the second device at least partially by a user, and wherein at least a portion of the another one or more instruction sets for operating the third device or at least an information related to the another one or more instruction sets for operating the third device is learned in another learning process that includes operating the third device at least partially by the user.

56. The first device of claim **46**, wherein the one or more inputs are further correlated with another one or more instruction sets for operating a third device, and wherein at least a portion of the one or more instruction sets for operating the second device or at least an information related to the one or more instruction sets for operating the second device is learned in a learning process that includes operating the second device at least partially by a user, and wherein at least a portion of the another one or more instruction sets for operating the third device or at least an

information related to the another one or more instruction sets for operating the third device is learned in another learning process that includes operating the third device at least partially by another user.

57. The first device of claim **46**, wherein the one or more inputs are further correlated with another one or more instruction sets for operating a third device.

58. The first device of claim **46**, wherein the one or more object representations of the generated circumstance representation represent one or more objects in the first device's surrounding, and wherein the first device's surrounding includes at least one of:

an area of interest around the first device, or

an area within a threshold distance from the first device.

59. The first device of claim **46**, wherein the one or more object representations of the generated circumstance representation include one or more information about one or more objects.

60. The first device of claim **46**, wherein the one or more object representations of the generated circumstance representation include one or more coordinates of one or more objects.

61. The first device of claim **46**, wherein the one or more object representations of the generated circumstance representation include at least one of: one or more information about one or more distances of one or more objects relative to the first device, or one or more information about one or more angles of one or more objects relative to the first device.

62. The first device of claim **46**, wherein the one or more object representations of the generated circumstance representation include one or more three dimensional representations of one or more objects.

63. The first device of claim **46**, wherein the one or more object representations of the generated circumstance representation include: one or more digital pictures that depict one or more objects, or one or more digital pictures that depict one or more representations of one or more objects.

64. The first device of claim **46**, wherein the machine readable code, when executed by the one or more processors, causes the one or more processors to further perform at least:

modifying the one or more instruction sets for operating the second device,

wherein the determining the one or more instruction sets for operating the second device at least by: the inputting the at least the portion of the generated circumstance representation into the one or more inputs, and the using the correlation between the one or more inputs and the one or more instruction sets for operating the second device includes determining the modified one or more instruction sets for operating the second device at least by: the inputting the at least the portion of the generated circumstance representation into the one or more inputs, and using the correlation between the one or more inputs and the modified one or more instruction sets for operating the second device, and

wherein the executing the one or more instruction sets for operating the second device includes executing the modified one or more instruction sets for operating the second device, and

wherein the first device autonomously performing the one or more operations defined by the one or more instruction sets for operating the second device includes the first device autonomously performing one or more operations defined by the modified one or more instruction sets for operating the second device.

65. The first device of claim **46**, wherein the machine readable code, when executed by the one or more processors, causes the one or more processors to further perform at least:

modifying: the one or more instruction sets for operating the second device, or a copy of the one or more instruction sets for operating the second device, and

wherein the executing the one or more instruction sets for operating the second device includes: executing the modified one or more instruction sets for operating the second device, or executing the modified the copy of the one or more instruction sets for operating the second device, and

wherein the first device autonomously performing the one or more operations defined by the one or more instruction sets for operating the second device includes: the first device autonomously performing one or more operations defined by the modified one or more instruction sets for operating the second device, or the first device autonomously performing one or more operations defined by the modified the copy of the one or more instruction sets for operating the second device.

66. The first device of claim **46**, wherein the machine readable code, when executed by the one or more processors, causes the one or more processors to further perform at least:

modifying: the generated circumstance representation, or a copy of the generated circumstance representation, and

wherein the inputting the at least the portion of the generated circumstance representation into the one or more inputs includes: inputting at least a portion of the modified the generated circumstance representation into the one or more inputs, or inputting at least a portion of the modified the copy of the generated circumstance representation into the one or more inputs.

67. The first device of claim **46**, wherein the one or more instruction sets for operating the second device in the knowledgebase are received from a server, and wherein the one or more instruction sets for operating the second device on the server are received from the second device.

68. The first device of claim **46**, wherein the knowledgebase is a neural network.

69. The first device of claim **46**, wherein the knowledgebase includes an artificial intelligence system.

70. The first device of claim **46**, wherein the one or more inputs are correlated with the one or more instruction sets for operating the second device using at least one or more connections, and wherein the using the correlation between the one or more inputs and the one or more instruction sets for operating the second device includes using at least one connection of the one or more connections between the one or more inputs and the one or more instruction sets for operating the second device.

71. The first device of claim **46**, wherein the one or more inputs are one or more input nodes, and wherein the knowledgebase further includes one or more output nodes that include the one or more instruction sets for operating the second device, and wherein the one or more input nodes are connected with the one or more output nodes using at least one or more connections.

72. The first device of claim **46**, wherein the knowledgebase further includes one or more outputs that include the one or more instruction sets for operating the second device, and wherein the correlation between the one or more inputs and the one or more instruction sets for operating the second device or an information related to the correlation between the one or more inputs and the one or more instruction sets for operating the second device is learned in a learning process that includes:

generating another circumstance representation, wherein the generated another circumstance representation represents another circumstance detected at least in part by one or more sensors of the second device;

obtaining or receiving the one or more instruction sets for operating the second device;

inputting at least a portion of the generated another circumstance representation into the one or more inputs; and

applying the one or more instruction sets for operating the second device to the one or more outputs.

73. The first device of claim **72**, wherein the learning process further includes:

back-propagating an information from the one or more outputs to the one or more inputs.

74. The first device of claim **46**, wherein the knowledgebase further includes one or more outputs that include the one or more instruction sets for operating the second device, and wherein the determining the one or more instruction sets for operating the second device is further performed at least by:

receiving the one or more instruction sets for operating the second device from the one or more outputs.

75. The first device of claim **46**, wherein the generated circumstance representation includes a representation of the first device.

76. The first device of claim **46**, wherein the one or more instruction sets for operating the second device include one or more information about one or more states of: the second device, or a portion of the second device.

77. The first device of claim **46**, wherein the one or more instruction sets for operating the second device include one or more representations of another one or more instruction sets for operating the second device.

78. The first device of claim **46**, wherein the one or more instruction sets for operating the second device are further for operating at least a third device.

79. The first device of claim **46**, wherein the generated circumstance representation represents the circumstance detected at least in part by the one or more sensors at a time.

80. The first device of claim **46**, wherein the generated circumstance representation represents the circumstance detected at least in part by the one or more sensors during a time period.

81. The first device of claim **46**, wherein the generated circumstance representation is a data structure that includes one or more data about the circumstance detected at least in part by the one or more sensors, and wherein the circumstance detected at least in part by the one or more sensors includes one or more objects detected at least in part by the one or more sensors: at a first time, or during a first time period.

82. The first device of claim **46**, wherein the one or more processors include: one or more computing devices, one or more electronic devices, or one or more microcontrollers, and wherein the one or more sensors include at least one of: one or more cameras, one or more microphones, one or more radars, one or more lidars, one or more sonars, one or more apparatuses for detecting circumstances, or one or more apparatuses for detecting objects or object properties, and wherein an instruction set of the one or more instruction sets for operating the second device includes at least one of: only one instruction, multiple instructions, one or more com-

mands, one or more keywords, one or more symbols, one or more operators, one or more variables, one or more parameters, one or more characters, one or more numbers, one or more values, one or more signals, one or more binary bits, one or more functions, one or more function references, one or more objects, one or more object references, one or more data structures, one or more data structure references, one or more states, one or more representations of one or more states, one or more inputs, one or more representations of one or more inputs, one or more codes, one or more data, or one or more information, and wherein the one or more object representations include: one object representation, multiple object representations, a collection of object representations, or a stream of collections of object representations, and wherein the one or more object representations include: one or more three dimensional representations of one or more objects, one or more digital pictures that depict one or more objects, one or more digital pictures that depict one or more representations of one or more objects, one or more information about one or more properties of one or more objects, or one or more computer representations of one or more objects, and wherein the at least the portion of the generated circumstance representation includes: one portion of the generated circumstance representation, multiple portions of the generated circumstance representation, or the entire generated circumstance representation, and wherein the at least the portion of the generated circumstance representation includes: one portion of an object representation of the one or more object representations of the generated circumstance representation, multiple portions of an object representation of the one or more object representations of the generated circumstance representation, multiple portions of multiple object representations of the one or more object representations of the generated circumstance representation, one object representation of the one or more object representations of the generated circumstance representation, multiple object representations of the one or more object representations of the generated circumstance representation, or the entire one or more object representations of the generated circumstance representation.

83. The first device of claim **46**, wherein at least a portion of the knowledgebase is stored in or on at least one of: at least one non-transitory machine readable medium of the one or more non-transitory machine readable media, another one or more non-transitory machine readable media, one or more volatile memories, one or more non-volatile memories, one or more storage devices, or one or more storage systems.

84. A system comprising:

means for accessing a knowledgebase that includes one or more inputs for inputting at least a portion of a circumstance representation, wherein the one or more inputs are correlated with one or more instruction sets for operating a first device;

means for generating a circumstance representation, wherein the generated circumstance representation represents a circumstance detected at least in part by one or more sensors of a second device;

means for determining the one or more instruction sets for operating the first device at least by: inputting at least a portion of the generated circumstance representation into the one or more inputs, and using a correlation between the one or more inputs and the one or more instruction sets for operating the first device; and

means for executing, at least in response to the determining, the one or more instruction sets for operating the first device, wherein the second device autonomously performs one or more operations defined by the one or more instruction sets for operating the first device.

* * * * *

Exhibit E

US011055583B1

(12) **United States Patent**
Cosic

(10) **Patent No.: US 11,055,583 B1**
(45) **Date of Patent: *Jul. 6, 2021**

(54) **MACHINE LEARNING FOR COMPUTING ENABLED SYSTEMS AND/OR DEVICES**

(71) Applicant: **Jasmin Cosic**, Miami, FL (US)

(72) Inventor: **Jasmin Cosic**, Miami, FL (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 67 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: **16/584,736**

(22) Filed: **Sep. 26, 2019**

Related U.S. Application Data

(63) Continuation of application No. 15/822,150, filed on Nov. 26, 2017, now Pat. No. 10,474,934.

(51) **Int. Cl.**

G06K 9/66	(2006.01)
G06N 5/02	(2006.01)
G06K 9/62	(2006.01)
G06T 7/70	(2017.01)
G06F 9/30	(2018.01)
G06N 20/00	(2019.01)
G06F 15/80	(2006.01)
G06F 15/78	(2006.01)

(52) **U.S. Cl.**
CPC ***G06K 9/66*** (2013.01); ***G06F 9/30076*** (2013.01); ***G06K 9/6202*** (2013.01); ***G06N 20/00*** (2019.01); ***G06T 7/70*** (2017.01); *G06F 15/78* (2013.01); *G06F 15/80* (2013.01); *G06N 5/022* (2013.01)

(58) **Field of Classification Search**
CPC .. G06K 9/00355; G06K 9/00389; G06K 9/66; G06T 2207/20081; G06F 3/002; G06F 3/005; G06F 3/017; G06N 5/022; G06N 5/025; G06N 20/00
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,370,707	A	1/1983	Phillips et al.	364/200
4,730,315	A	3/1988	Saito et al.	371/19
4,860,203	A	8/1989	Corrigan et al.	364/300
5,560,011	A	9/1996	Uyama	395/700
5,602,982	A	2/1997	Judd et al.	395/326
6,026,234	A	2/2000	Hanson et al.	395/704
6,088,731	A	7/2000	Kiraly et al.	709/229
6,106,299	A	8/2000	Ackermann et al.	434/81
6,126,330	A	10/2000	Knight	395/704
6,314,558	B1	11/2001	Angel et al.	717/4
6,408,262	B1	6/2002	Leerberg et al.	703/2
6,643,842	B2	11/2003	Angel et al.	717/130
6,728,689	B1	4/2004	Drissi et al.	706/14
6,735,632	B1	5/2004	Kiraly et al.	709/229
6,754,631	B1	6/2004	Din	704/270

(Continued)

OTHER PUBLICATIONS

About Event Tracing, retrieved from <URL: http://msdn.microsoft.com/en-us/library/aa363668(d=default,l=en-us,v=vs.85).aspx> on Jan. 12, 2014, 2 pages.

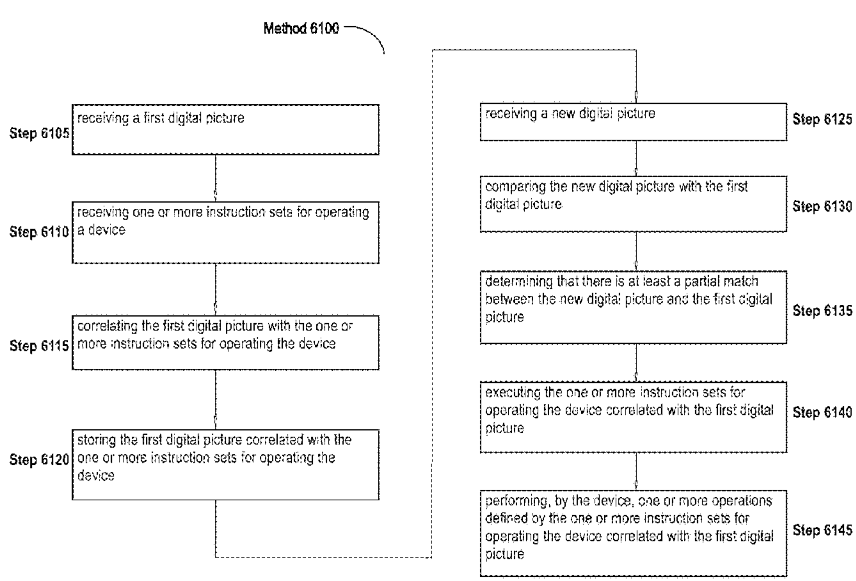
(Continued)

Primary Examiner — Andrew W Johns

(57) **ABSTRACT**

Aspects of the disclosure generally relate to computing enabled systems and/or devices and may be generally directed to machine learning for computing enabled systems and/or devices. In some aspects, the system captures one or more digital pictures, receives one or more instruction sets, and learns correlations between the captured pictures and the received instruction sets.

34 Claims, 41 Drawing Sheets



(56) **References Cited**

U.S. PATENT DOCUMENTS

6,801,912	B2	10/2004	Moskowitz et al.	707/10
6,842,877	B2	1/2005	Robarts et al.	715/708
6,915,105	B2	7/2005	Masuda	434/350
6,973,446	B2	12/2005	Mamitsuka et al.	706/50
7,017,153	B2	3/2006	Gouriou et al.	717/158
7,052,277	B2	5/2006	Kellman	434/118
7,113,946	B2	9/2006	Cosic	707/8
7,117,225	B2	10/2006	Cosic	707/102
7,222,127	B1	5/2007	Bem et al.	707/102
7,240,335	B2	7/2007	Angel et al.	717/130
7,249,349	B2	7/2007	Hundt et al.	717/130
7,424,705	B2	9/2008	Lewis et al.	717/148
7,478,371	B1	1/2009	Gove	717/128
7,484,205	B2	1/2009	Venkatapathy	717/158
7,565,340	B2	7/2009	Herlocker et al.	706/52
7,721,218	B2	5/2010	Awe et al.	715/762
7,765,537	B2	7/2010	Havin et al.	717/166
7,797,259	B2	9/2010	Jiang et al.	706/21
7,840,060	B2	11/2010	Podilchuk	382/155
7,925,984	B2	4/2011	Awe et al.	715/762
7,987,144	B1	7/2011	Drissi et al.	706/20
8,005,828	B2	8/2011	Buchner et al.	707/728
8,019,699	B2	9/2011	Baxter	706/12
8,078,556	B2	12/2011	Adi et al.	706/16
8,090,669	B2	1/2012	Shahani et al.	706/14
8,137,112	B2	3/2012	Woolf et al.	434/323
8,166,463	B2	4/2012	Gill et al.	717/130
8,195,674	B1	6/2012	Bem et al.	707/749
8,196,119	B2	6/2012	Gill et al.	717/130
8,244,730	B2	8/2012	Gupta	707/737
8,251,704	B2	8/2012	Woolf et al.	434/322
8,261,199	B2	9/2012	Cradick et al.	715/757
8,266,608	B2	9/2012	Hecht et al.	717/158
8,335,805	B2	12/2012	Cosic	707/805
8,356,026	B2	1/2013	Heimendinger	707/713
8,364,612	B2	1/2013	Van Gael et al.	706/12
8,386,401	B2	2/2013	Virkar et al.	706/12
8,397,227	B2	3/2013	Fan et al.	717/166
8,417,740	B2	4/2013	Cosic	707/805
8,464,225	B2	6/2013	Greifeneder	717/130
8,549,359	B2	10/2013	Zheng	714/38.11
8,572,035	B2	10/2013	Cosic	707/625
8,589,414	B2	11/2013	Waite et al.	707/749
8,595,154	B2	11/2013	Breckenridge et al.	706/12
8,655,260	B2	2/2014	Yueh et al.	434/351
8,655,900	B2	2/2014	Cosic	707/760
8,667,472	B1	3/2014	Molinari	717/130
8,996,432	B1	3/2015	Fu	G06N 99/005
9,047,324	B2	6/2015	Cosic	G06F 17/30289
9,158,813	B2	10/2015	Ntoulas et al.	G06F 17/30451
9,268,454	B2	2/2016	Hamilton, II et al.	G06F 3/0481
9,282,309	B1	3/2016	Cosic	H04N 9/87
9,298,749	B2	3/2016	Cosic	G06F 17/30289
9,305,216	B1	4/2016	Mishra	G06K 9/00624
9,367,806	B1	6/2016	Cosic	G06N 5/022
9,443,192	B1	9/2016	Cosic	G06N 3/088
9,488,984	B1 *	11/2016	Williams et al.	G06K 9/2036
9,582,762	B1	2/2017	Cosic	G06N 5/04
9,595,294	B2	3/2017	Cosic	G11B 27/036
9,697,869	B2	7/2017	Cosic	G11B 27/036
9,864,933	B1 *	1/2018	Cosic	G06K 9/52
10,162,740	B1	12/2018	Setty et al.	G06F 1/3688
10,474,934	B1 *	11/2019	Cosic	G06K 9/6262
2003/0026588	A1	2/2003	Elder et al.	386/46
2003/0065662	A1	4/2003	Cosic	707/9
2004/0117771	A1	6/2004	Venkatapathy	717/130
2004/0194017	A1	9/2004	Cosic	707/501.1
2004/0249774	A1	12/2004	Caid et al.	706/14
2004/0267521	A1	12/2004	Cutler et al.	704/202
2005/0149517	A1	7/2005	Cosic	707/3
2005/0149542	A1	7/2005	Cosic	707/100
2005/0240412	A1	10/2005	Fujita	704/270
2005/0245303	A1	11/2005	Graepel et al.	463/1
2005/0289105	A1	12/2005	Cosic	707/1
2006/0047612	A1	3/2006	Stanley et al.	706/16
2006/0184410	A1	8/2006	Ramamurthy et al.	705/8
2006/0190930	A1	8/2006	Hecht et al.	717/28
2006/0265406	A1	11/2006	Chkodrov et al.	707/100
2007/0006159	A1	1/2007	Hecht et al.	717/124
2007/0050606	A1	3/2007	Ferren et al.	712/227
2007/0050719	A1	3/2007	Lui et al.	715/762
2007/0058856	A1	3/2007	Boregowda et al.	382/159
2007/0061735	A1	3/2007	Hoffberg et al.	715/744
2007/0106633	A1	5/2007	Reiner	707/1
2008/0144893	A1	6/2008	Guo et al.	382/118
2008/0215508	A1	9/2008	Hanneman et al.	706/11
2008/0254429	A1	10/2008	Woolf et al.	434/322
2008/0281764	A1	11/2008	Baxter	706/12
2008/0288259	A1	11/2008	Chambers et al.	704/275
2009/0044113	A1	2/2009	Jones et al.	715/707
2009/0067727	A1	3/2009	Hirohata	382/201
2009/0110061	A1	4/2009	Lee et al.	375/240.02
2009/0131152	A1	5/2009	Busse	463/23
2009/0136095	A1	5/2009	Marcon et al.	382/118
2009/0141969	A1	6/2009	Yu et al.	382/157
2009/0222388	A1	9/2009	Hua et al.	706/12
2009/0287643	A1	11/2009	Corville et al.	707/3
2009/0324010	A1	12/2009	Hou	382/103
2010/0023541	A1	1/2010	Cosic	707/101
2010/0033780	A1	2/2010	Gitter	359/9
2010/0063949	A1	3/2010	Eaton et al.	706/14
2010/0082536	A1	4/2010	Cosic	7/610
2010/0114746	A1	5/2010	Bobbitt et al.	705/35
2010/0138370	A1	6/2010	Wu et al.	706/12
2010/0241595	A1	9/2010	Felsher	705/400
2010/0278420	A1	11/2010	Shet et al.	382/156
2011/0007079	A1 *	1/2011	Perez et al.	H04N 21/4532 345/473
2011/0030031	A1	2/2011	Lussier et al.	726/1
2011/0085734	A1	4/2011	Berg et al.	382/195
2011/0218672	A1	9/2011	Maisonnier et al.	700/246
2011/0270794	A1	11/2011	Drory et al.	706/52
2012/0150773	A1	6/2012	DiCorpo et al.	706/12
2012/0167057	A1	6/2012	Schmich et al.	717/130
2012/0192096	A1	7/2012	Bowman et al.	715/780
2012/0284026	A1	11/2012	Cardillo et al.	704/246
2012/0290347	A1	11/2012	Elazouni et al.	705/7.12
2013/0007532	A1	1/2013	Miller et al.	714/45
2013/0156345	A1	6/2013	Shmunk	G06T 3/4053
2013/0159021	A1	6/2013	Felsher	G06F 19/322
2013/0218932	A1	8/2013	Cosic	G06F 17/30289
2013/0226974	A1	8/2013	Cosic	G06F 17/30289
2013/0238533	A1	9/2013	Virkar et al.	G06N 99/005
2013/0278501	A1	10/2013	Bulzacki	G06F 3/017
2013/0278631	A1	10/2013	Border et al.	G02B 27/017
2013/0329946	A1	12/2013	Archibald et al.	382/103
2014/0052717	A1	2/2014	Franks et al.	G06F 17/30867
2014/0075249	A1	3/2014	Sato et al.	G06F 11/3466
2014/0161250	A1	6/2014	Leeds et al.	H04M 1/656
2014/0177946	A1	6/2014	Lim et al.	G06K 9/00369
2014/0207580	A1	7/2014	Minnis et al.	G06Q 30/0269
2014/0211988	A1	7/2014	Fan et al.	G06K 9/3241
2014/0278754	A1	9/2014	Cronin et al.	G06Q 30/0201
2015/0006171	A1	1/2015	Westby et al.	G10L 15/26
2015/0039304	A1	2/2015	Wein	G10L 25/78
2015/0055821	A1	2/2015	Fotland	G06K 9/3241
2015/0264306	A1	9/2015	Marilly et al.	H04N 7/147
2015/0269156	A1	9/2015	Awadallah et al.	G06F 17/3053
2015/0269415	A1	9/2015	Gelbman	G06K 9/00214
2015/0310041	A1	10/2015	Kier et al.	G06F 17/30259
2015/0324685	A1	11/2015	Bohn et al.	G06N 3/04
2015/0339213	A1	11/2015	Lee et al.	G06F 11/3664
2016/0063118	A1	3/2016	Campbell et al.	G06F 17/30867
2016/0077166	A1	3/2016	Morozov et al.	G01R 33/0286
2016/0140999	A1	5/2016	Cosic	G11B 27/036
2016/0142650	A1	5/2016	Cosic	H04N 5/2624
2016/0167226	A1	6/2016	Schnittman	B25J 9/163
2016/0243701	A1	8/2016	Gildert et al.	B25J 9/1656
2016/0246819	A1	8/2016	Cosic	G06F 17/30289
2016/0246850	A1	8/2016	Cosic	G06F 17/30528
2016/0246868	A1	8/2016	Cosic	G06F 17/30595

(56) **References Cited**

U.S. PATENT DOCUMENTS

2016/0274187	A1	9/2016	Menon et al.	G01R 31/31718
2016/0292185	A1	10/2016	Cosic	G06F 17/30289
2016/0328480	A1	11/2016	Owens et al.	G01F 17/30867
2017/0161958	A1	6/2017	Eilat	G06T 19/006
2018/0068564	A1*	3/2018	Tanigawa et al.	G08G 1/143
2018/0101146	A1	4/2018	Hariharan et al.	G05B 13/026
2018/0120813	A1	5/2018	Coffman et al.	G05B 19/4097
2019/0034765	A1*	1/2019	Kaehler et al.	G02B 27/0172
2019/0107839	A1	4/2019	Parashar et al.	G05D 1/0219

OTHER PUBLICATIONS

Add dynamic Java code to your application, retrieved from <URL: http://www.javaworld.com/article/2071777/design-patterns/add-dynamic-java-code-to-your-application.html> on May 13, 2014, 6 pages.
BCEL, retrieved from <URL: http://commons.apache.org> on May 13, 2014, 2 pages.
Artificial intelligence, retrieved from <URL: http://wikipedia.com> on May 21, 2014, 28 pages.
Artificial neural network, retrieved from <URL: http://wikipedia.com> on May 21, 2014, 12 pages.
Branch (computer science), retrieved from <URL: http://wikipedia.com> on May 6, 2014, 2 pages.
Branch trace, retrieved from <URL: http://wikipedia.com> on May 28, 2014, 2 pages.
C Function Conventions and the Stack, retrieved from <URL: http://www.csee.umbc.edu/~chang/cs313.s02/stack.shtml> on Jan. 11, 2014, 4 pages.
Call stack, retrieved from <URL: http://wikipedia.com> on Jan. 11, 2014, 5 pages.
Program compilation and execution flow, retrieved from <URL: http://cs.stackexchange.com/questions/6187/program-compilation-and-execution-flow> on Jan. 9, 2014, 2 pages.
Control unit, retrieved from <URL: http://wikipedia.com> on May 25, 2014, 2 pages.
Create dynamic applications with javax.tools, retrieved from <URL: http://www.ibm.com/developerworks/library/j-jcomp/> on May 13, 2014, 11 pages.
Creating a Debugging and Profiling Agent with JVMTI, retrieved from <URL: http://www.oracle.com/technetwork/articles/java/jvmti-136367.html> on Jan. 11, 2014, 18 pages.
Decision tree learning, retrieved from <URL: http://wikipedia.com> on Jun. 24, 2014, 5 pages.
Dynamic Bytecode Instrumentation, retrieved from <URL: http://www.drdobbs.com/tools/dynamic-bytecode-instrumentation/184406433> on Apr. 26, 2014, 6 pages.
Dynamic programming language, retrieved from <URL: http://wikipedia.com> on May 7, 2014, 4 pages.
Dynamic recompilation, retrieved from <URL: http://wikipedia.com> on May 19, 2014, 4 pages.
Eval( ), retrieved from <URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/eval> on May 7, 2014, 13 pages.
Function, retrieved from <URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Function> on May 7, 2014, 11 pages.
FunctionEnter2 Function, retrieved from <URL: https://msdn.microsoft.com/en-us/library/aa964981(d=default,l=en-us,v=vs.110).aspx> on Mar. 19, 2014, 2 pages.
FunctionLeave2 Function, retrieved from <URL: https://msdn.microsoft.com/en-us/library/aa964942(d=default,l=en-us,v=vs.110).aspx> on Mar. 19, 2014, 1 pages.
Heuristic, retrieved from <URL: http://wikipedia.com> on May 22, 2014, 5 pages.
How to: Configure Trace Switches, retrieved from <URL: https://msdn.microsoft.com/en-us/library/vstudio/t06xyy08(d=default,l=en-us,v=vs.100).aspx> on Mar. 19, 2014, 3 pages.
Indirect branch, retrieved from <URL: http://wikipedia.com> on May 5, 2014, 1 pages.
Instruction cycle, retrieved from <URL: http://wikipedia.com> on Apr. 30, 2014, 2 pages.
Instruction set, retrieved from <URL: http://wikipedia.com> on Apr. 29, 2014, 6 pages.
Interrupt, retrieved from <URL: http://wikipedia.com> on May 15, 2014, 5 pages.
Introduction to Instrumentation and Tracing, retrieved from <URL: http://msdn.microsoft.com/en-us/library/aa983649(d=default,l=en-us,v=vs.71).aspx> on Jan. 10, 2014, 2 pages.
Java bytecode, retrieved from <URL: http://wikipedia.com> on May 27, 2014, 4 pages.
Package java.util.logging, retrieved from <URL: http://docs.oracle.com/javase/1.5.0/docs/api/java/util/logging/package-summary.html> on Mar. 19, 2014, 2 pages.
Javac—Java programming language compiler, retrieved from <URL: http://docs.oracle.com/javase/7/docs/technotes/tools/windows/javac.html> on May 13, 2014, 12 pages.
Interface JavaCompiler, retrieved from <URL: http://wikipedia.com> on May 13, 2014, 4 pages.
Javassist, retrieved from <URL: http://wikipedia.com> on May 13, 2014, 2 pages.
Just-in-time compilation, retrieved from <URL: http://wikipedia.com> on May 19, 2014, 5 pages.
JVM Tool Interface, retrieved from <URL: http://docs.oracle.com/javase/7/docs/platform/jvmti/jvmti.html> on Apr. 26, 2014, 127 pages.
Machine code, retrieved from <URL: http://wikipedia.com> on Apr. 30, 2014, 3 pages.
Metaprogramming, retrieved from <URL: http://wikipedia.com> on May 8, 2014, 3 pages.
Microsoft.VisualBasic.Logging Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/ms128122(d=default,l=en-us,v=vs.110).aspx> on Jan. 13, 2014, 1 pages.
Pin (computer program), retrieved from <URL: http://wikipedia.com> on May 6, 2014, 3 pages.
Processor register, retrieved from <URL: http://wikipedia.com> on Apr. 29, 2014, 4 pages.
Profiling (computer programming), retrieved from <URL: http://wikipedia.com> on Jan. 10, 2014, 4 pages.
Profiling Overview, retrieved from <URL: http://msdn.microsoft.com/en-us/library/bb384493(d=default,l=en-us,v=vs.110).aspx> on Jan. 12, 2014, 6 pages.
Program counter, retrieved from <URL: http://wikipedia.com> on Apr. 24, 2014, 2 pages.
Ptrace, retrieved from <URL: http://wikipedia.com> on Apr. 24, 2014, 2 pages.
Reflection (computer programming), retrieved from <URL: http://wikipedia.com> on May 7, 2014, 5 pages.
Class Runtime, retrieved from <URL: http://docs.oracle.com/javase/7/docs/api/java/lang/Runtime.html> on Jan. 10, 2014, 10 pages.
Self-modifying code, retrieved from <URL: http://wikipedia.com> on Jan. 12, 2014, 6 pages.
Subjective logic, retrieved from <URL: http://wikipedia.com> on May 22, 2014, 5 pages.
System.CodeDom.Compiler Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/z6b99ydt(d=default,l=en-us,v=vs.110).aspx> on May 13, 2014, 2 pages.
System.Reflection.Emit Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/xd5fw18y(d=default,l=en-us,v=vs.110).aspx> on May 13, 2014, 2 pages.
About | OpenCV, retrieved from <URL: http://opencv.org/about.html> on Dec. 13, 2014, 1 pages.
Animetrics, Inc. | 3D Facial Recognition, retrieved from <URL: http://animetrics.com/> on Dec. 13, 2014, 2 pages.
Bag-of-words model, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 2 pages.
Beier-Neely morphing algorithm, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 1 pages.
Computer vision, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 7 pages.

(56) **References Cited**

OTHER PUBLICATIONS

Convolutional neural network, retrieved from <URL: http://wikipedia.com> on Nov. 11, 2015, 5 pages.
Decimation (signal processing), retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 3 pages.
Digital image processing, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 3 pages.
Dissolve (filmmaking), retrieved from <URL: http://wikipedia.com> on Nov. 11, 2015, 2 pages.
Facial recognition system, retrieved from <URL: http://wikipedia.com> on Nov. 11, 2015, 6 pages.
Feature detection (computer vision), retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 3 pages.
Feature extraction, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 2 pages.
Gesture recognition, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 5 pages.
GrabCut, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 1 pages.
Image processing, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 2 pages.
Image segmentation, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 12 pages.
Image warping, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 2 pages.
Inbetweening, retrieved from <URL: http://wikipedia.com> on Nov. 11, 2015, 2 pages.
Interpolation, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 5 pages.
Language model, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 4 pages.
List of speech recognition software, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 4 pages.
Livewire Segmentation Technique, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 2 pages.
Morphing, retrieved from <URL: http://wikipedia.com> on Nov. 11, 2015, 2 pages.
Motion estimation, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 2 pages.
Muse: Face Morph || Mesh Warping, retrieved from <URL: http://alexwolfe.blogspot.com/2011/10/face-morph-mesh-warping.html> on Nov. 19, 2015, 2 pages.
N-gram, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 5 pages.
Optical flow, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 4 pages.
Outline of object recognition, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 7 pages.
Phonotactics, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 3 pages.
Recurrent neural network, retrieved from <URL: http://wikipedia.com> on Nov. 11, 2015, 5 pages.
Sample rate conversion, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 2 pages.
Simple interactive object extraction, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 2 pages.
Speech recognition, retrieved from <URL: http://wikipedia.com> on Nov. 11, 2015, 12 pages.
Speech segmentation, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 4 pages.
Welcome to recognize-speech.com, retrieved from <URL: http://recognize-speech.com/> on Oct. 18, 2015, 1 pages.
Introduction Speech, retrieved from <URL: http://recognize-speech.com/speech> on Oct. 18, 2015, 1 pages.
Preprocessing, retrieved from <URL: http://recognize-speech.com/preprocessing> on Oct. 18, 2015, 4 pages.
Feature Extraction, retrieved from <URL: http://recognize-speech.com/feature-extraction> on Oct. 18, 2015, 3 pages.
Acoustic model, retrieved from <URL: http://recognize-speech.com/acoustic-model> on Oct. 18, 2015, 2 pages.
Video content analysis, retrieved from <URL: http://wikipedia.com> on Nov. 19, 2015, 2 pages.
Video tracking, retrieved from <URL: http://wikipedia.com> on Nov. 1, 2015, 2 pages.
Andrej Karpathy, Thomas Leung, George Toderici, Rahul Sukthankar, Sanketh Shetty, Li Fei-Fei, Large-scale Video Classitication with Convolutional Neural Networks, Apr. 14, 2014, 8 pages, Stanford University.
Karen Simonyan, Andrew Zisserman, Two-Stream Convolutional Networks for Action Recognition in Videos, Nov. 13, 2014, 11 pages, University of Oxford.
Tracing (software), retrieved from <URL: http://wikipedia.com> on Jan. 10, 2014, 3 pages.
Tree (data structure), retrieved from <URL: http://wikipedia.com> on Jun. 24, 2014, 6 pages.
PTRACE(2), retrieved from <URL: http://unixhelp.ed.ac.uk/CGI/man-cgi?ptrace> on Mar. 19, 2014, 5 pages.
Wevtutil, retrieved from <URL: http://technet.microsoft.com/en-us/library/cc732848(d=default,l=en-us,v=ws.11).aspx> on Apr. 28, 2014, 5 pages.
Intel Processor Trace, retrieved from <URL: https://software.intel.com/en-us/blogs/2013/09/18/processor-tracing> on Apr. 28, 2014, 3 pages.
Younghoon Jung, Java Dynamics Reflection and a lot more, Oct. 10, 2012, 55 pages, Columbia University.
Amitabh Srivastava, Alan Eustace, ATOM a System for Building Customized Program Analysis Tools, May 3, 2004, 12 pages.
Mathew Smithson, Kapil Anand, Aparna Kotha, Khaled Elwazeer, Nathan Giles, Rajeev Barua, Binary Rewriting without Relocation Information, Nov. 10, 2010, 11 pages, University of Maryland.
Marek Olszewski, Keir Mierte, Adam Czajkowski, Angela Demle Brown, JIT Instrumentation—A Novel Approach to Dynamically Instrument Operating Systems, Feb. 12, 2007, 14 pages, University of Toronto.
Chen et al. Case-Based Reasoning System and Artificial Neural Networks: A Review Neural Comput & Applic (2001) 10: pp. 264-276, 13 pages.
John J. Grefenstette, Connie Loggia Ramsey, Alan C. Schultz, Learning Sequential Decision Rules Using Simulation Models and Competition, 1990, Navy Center for Applied Research in Artificial Intelligence, Naval Research Laboratory, Washington, DC, 27 pages.
Alan C. Schultz, John J. Grefenstette, Using a Genetic Algorithm to Learn Behaviors for Autonomous Vehicles, 1992, Navy Center for Applied Research in Artificial Intelligence, Naval Research Laboratory, Washington, DC, 12 pages.
Koppula et al., "Anticipating human activities using object affordances for reactive robotic response", IEEE TRAMI 2016, published May 5, 2015, 16 pages.
Orme, "System design tips for entry level smartphones—part 3", found online at "https://community.arm.com/processors/b/blog/posts/system-design-tips-for-entry-level-smartphones---part-3", Oct. 21, 2013, 12 pages.
Nowostawski, Mariusz et al., "Dynamic Demes Parallel Genetic Algorithm", May 13, 1999, 6 pages.
Stack Overflow, How can I quantify difference between two images?, accessed Aug. 2, 2017 at https://stackoverflow.com/questions/189943/how-can-i-quantify-difference-between-two-images, 8 pages.
Skolicki, Zbigniew et al., "The Influence of Migration Sizes and Intervals on Island Models", Jun. 29, 2005, 8 pages.
Vahid Lari, et al., "Decentralized dynamic resource management support for massively parallel processor arrays", Sep. 11, 2011, 8 pages.
Vahid Lari, et al., "Distributed resource reservation in massively parallel processor arrays", May 16, 2011, 4 pages.
Mohsen Hayati and Yazdan Shirvany, "Artificial Neural Network Approach for Short Term Load Forecasting for Illam Region", Jan. 2007, 5 pages.
Hasim Sak, Andrew Senior, and Francoise Beaufays, "Long Short-Term Memory Recurrent Neural Network Architectures for Large Scale Acoustic Modeling", Jan. 2014, 5 pages.
Jorg Walter and Klaus Schulten, "Implementation of self-organizing neural networks for visuo-motor control of an industrial robot", Jan. 1993, 10 pages.

(56) **References Cited**

OTHER PUBLICATIONS

Heikki Hyotyniemi and Aamo Lehtola, "A Universal Relation Database Interface for Knowledge Based Systems", Apr. 1991, 5 pages.
Mrissa, Michael, et al. "An avatar architecture for the web of things." IEEE Internet Computing 19.2 (2015): 30-38., 9 pages.
Luck, Michael, and Ruth Aylett. "Applying artificial intelligence to virtual reality: Intelligent virtual environments." Applied Artificial Intelligence 14.1 (2000): 3-32., 30 pages.
Terdjimi, Mehdi, et al. "An avatar-based adaptation workflow for the web of things." Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), 2016 IEEE 25th International Conference on. IEEE, 2016., 6 pages.
Bogdanovych, Anton, et al. "Authentic interactive reenactment of cultural heritage with 3D virtual worlds and artificial intelligence." Applied Artificial Intelligence 24.6 (2010): 617-647., 32 pages.
Hernandez, Marco E. Perez, and Stephan Reiff-Marganiec. "Autonomous and self controlling smart objects for the future internet." Future Internet of things and cloud (FiCloud), 2015 3rd international conference on. IEEE, 2015., 8 pages.
Medini et al., "Building a Web of Things with Avatars", Managing the Web of Things (2017), 30 pages.
Zhang, Yiying, Lei Guo, and Nicolas D. Georganas. "AGILE: An architecture for agent-based collaborative and interactive virtual environments." Proc. Workshop on Application Virtual Reality Technologies for Future Telecommunication System, IEEE Globecom '2000 Conference. 2000. (Year: 2000), 9 pages.
Eladhari, Mirjam P., "Semi-Autonomous Avatars in Virtual Game Worlds," Pre-conference to the ECREA 2010—3rd European Communication Conference, Avatars and Humans. Representing Users in Digital Games, Hamburg, Germany, Oct. 2010, Available from: Jan. 21, 2012 (Year: 2012), 8 pages.

* cited by examiner

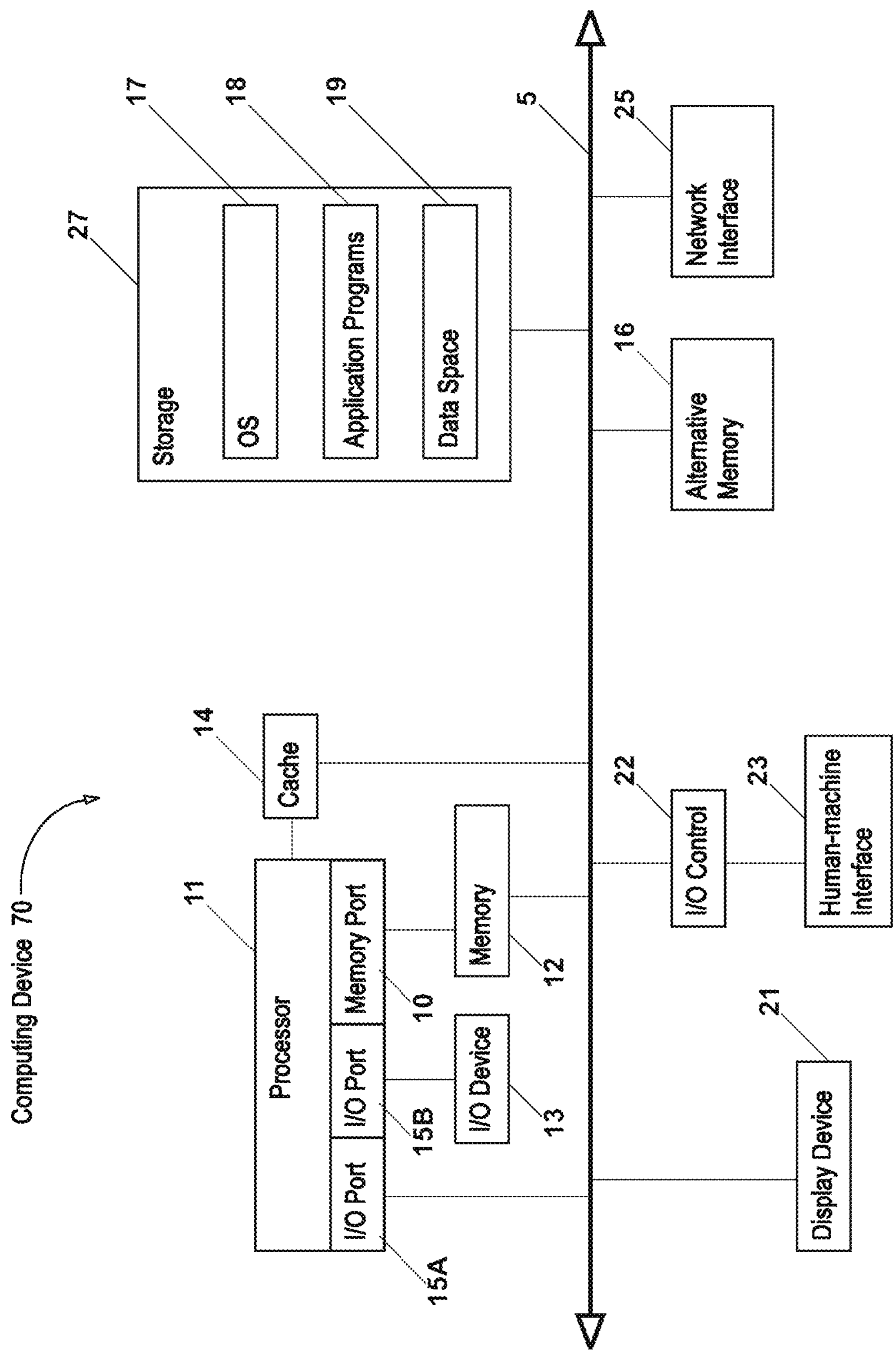

Computing Device 70
Storage
OS
Application Programs
Data Space
Network Interface
Alternative Memory
Cache
Processor
Memory Port
I/O Port
I/O Port
Memory
I/O Device
I/O Control
Human-machine Interface
Display Device
27
17
18
19
5
25
16
14
11
22
23
12
10
15B
13
21
15A

FIG. 1

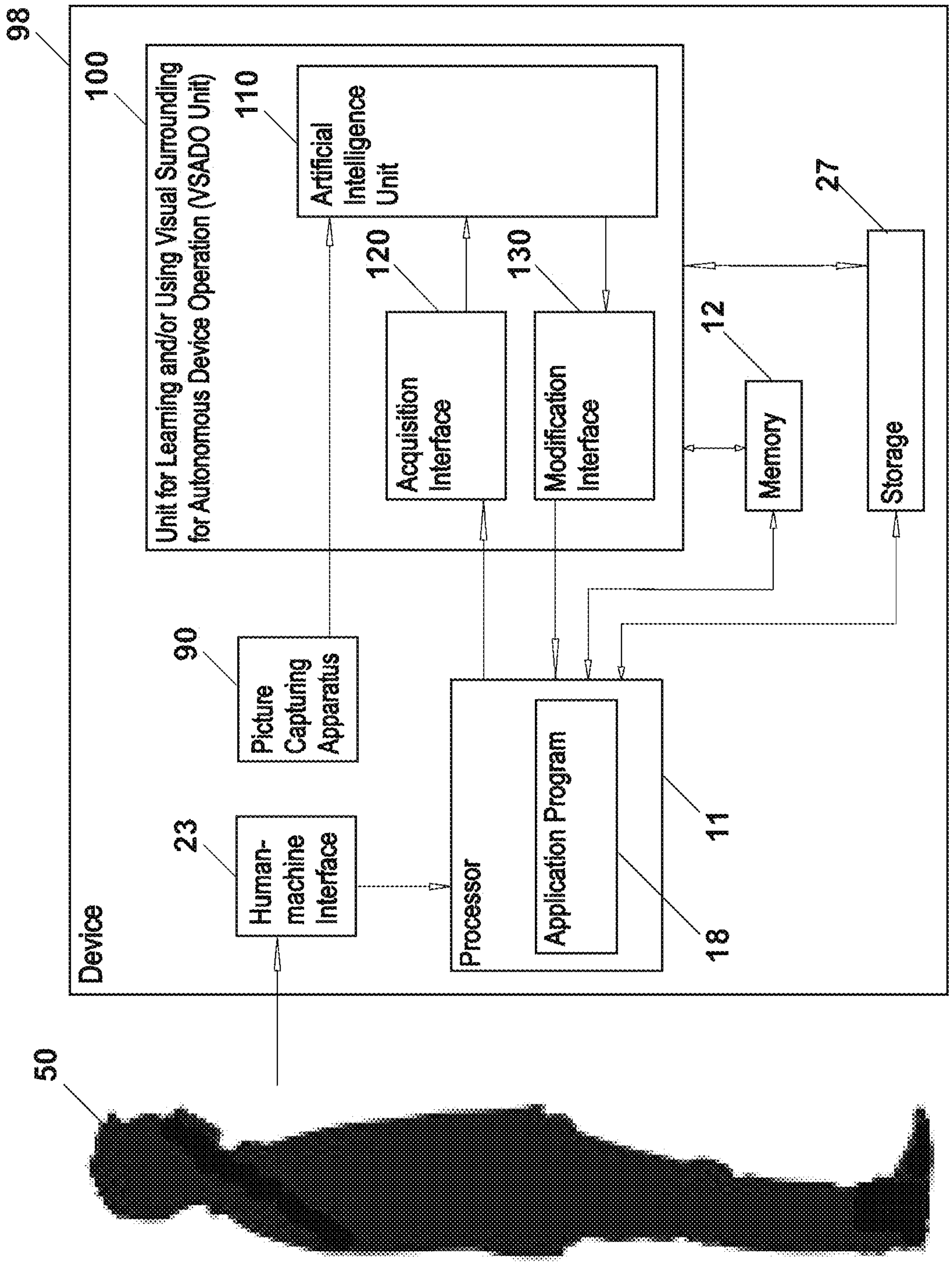

98
Device
100
Unit for Learning and/or Using Visual Surrounding for Autonomous Device Operation (VSADO Unit)
110
Artificial Intelligence Unit
120
Acquisition Interface
130
Modification Interface
12
Memory
27
Storage
90
Picture Capturing Apparatus
23
Human-machine Interface
Processor
Application Program
11
18
50

FIG. 2

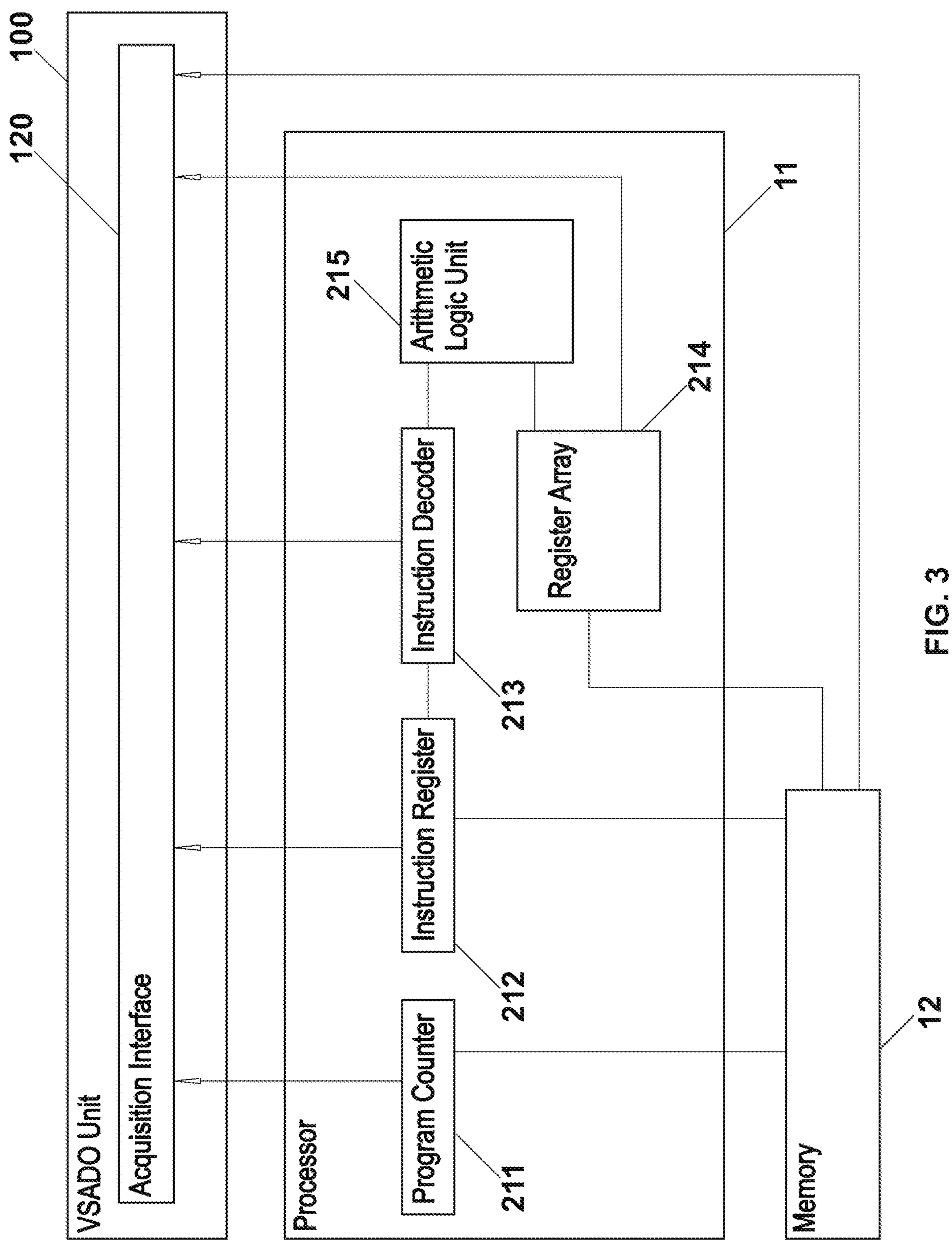

100
120
VSADO Unit
Acquisition Interface
Processor
11
Program Counter
211
Instruction Register
212
Instruction Decoder
213
Register Array
214
Arithmetic
Logic Unit
215
Memory
12

FIG. 3

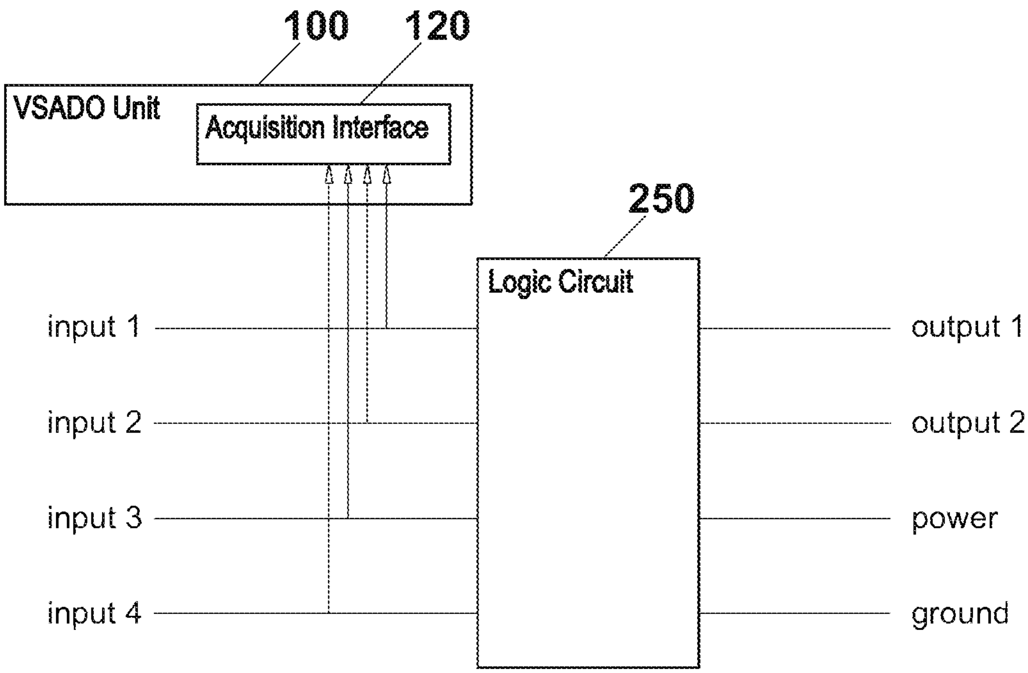

100
120
VSADO Unit
Acquisition Interface
250
Logic Circuit
input 1
input 2
input 3
input 4
output 1
output 2
power
ground

FIG. 4A

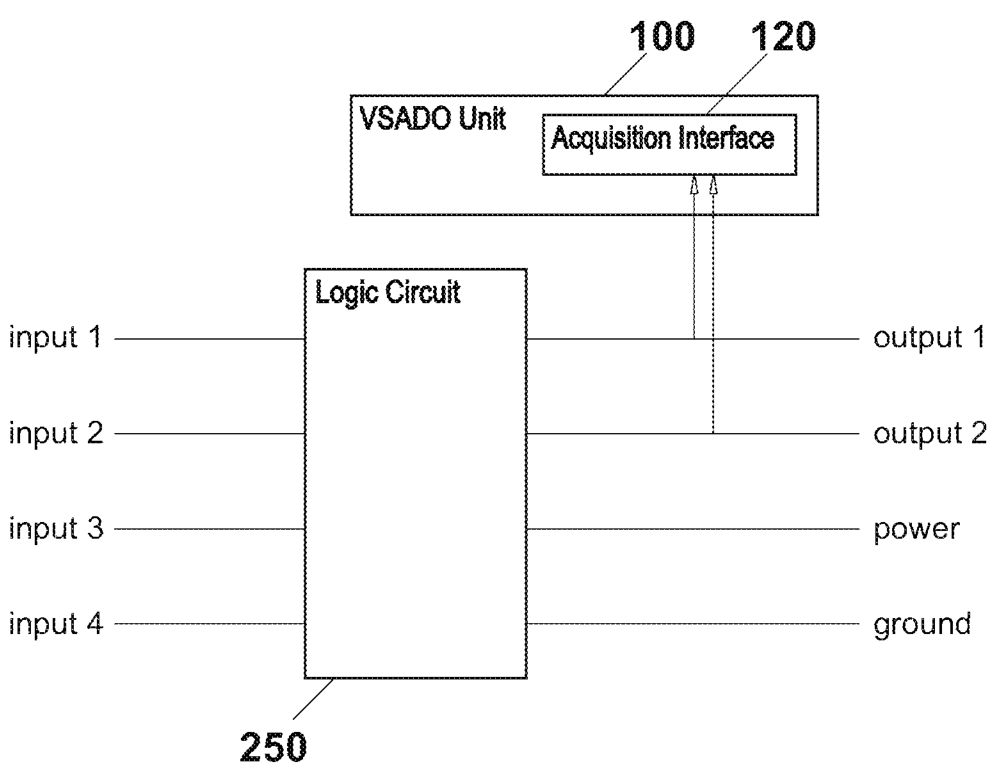

100
120
VSADO Unit
Acquisition Interface
Logic Circuit
input 1
input 2
input 3
input 4
output 1
output 2
power
ground
250

FIG. 4B

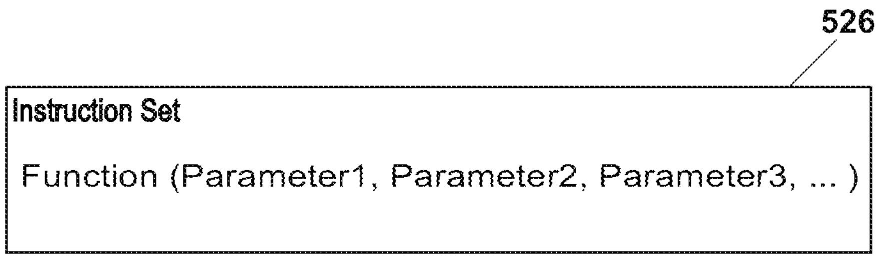


FIG. 5A

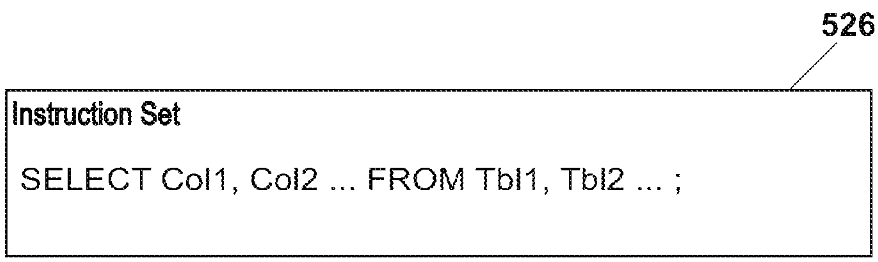


FIG. 5B

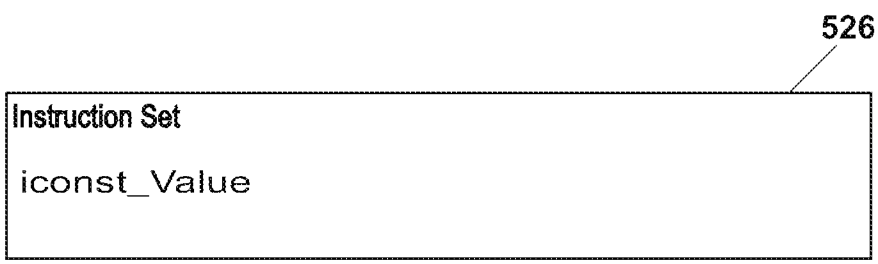


FIG. 5C

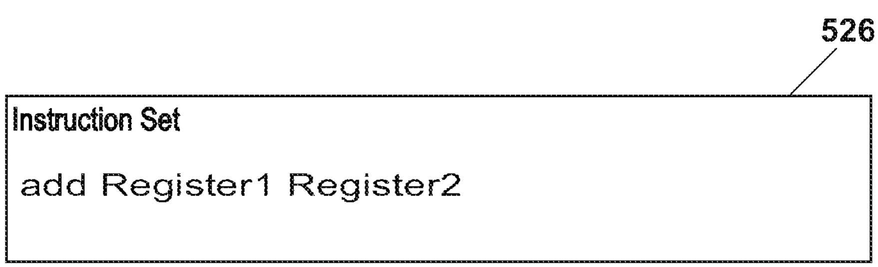


FIG. 5D

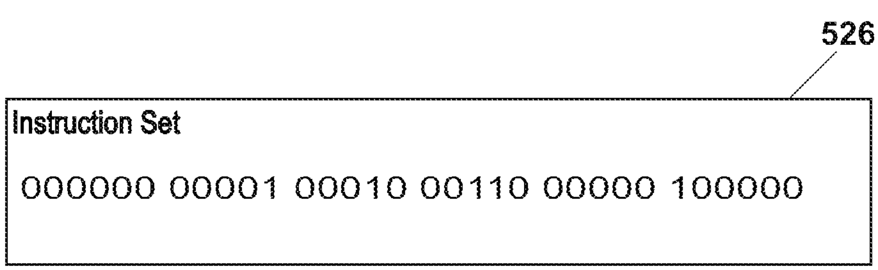


FIG. 5E

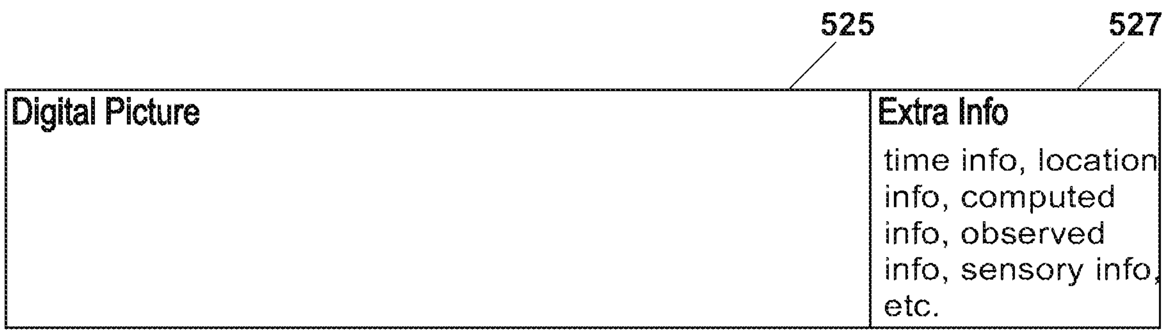

525
527
Digital Picture
Extra Info
time info, location info, computed info, observed info, sensory info, etc.

FIG. 6A

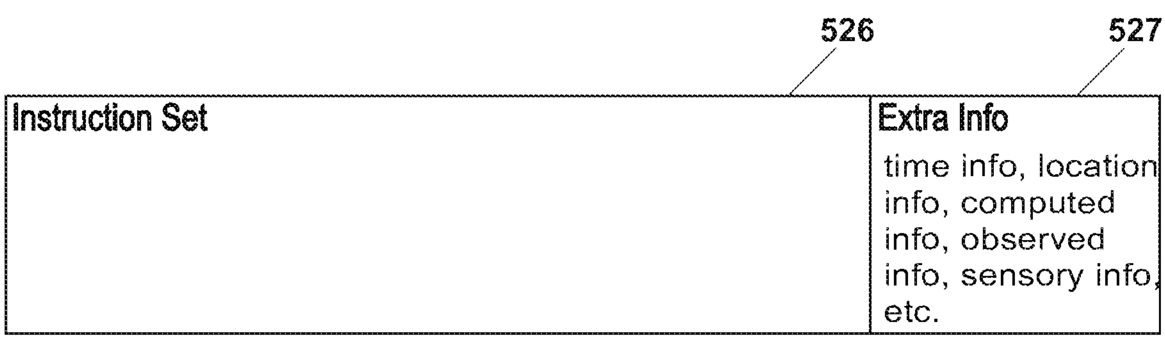

526
527
Instruction Set
Extra Info
time info, location info, computed info, observed info, sensory info, etc.

FIG. 6B

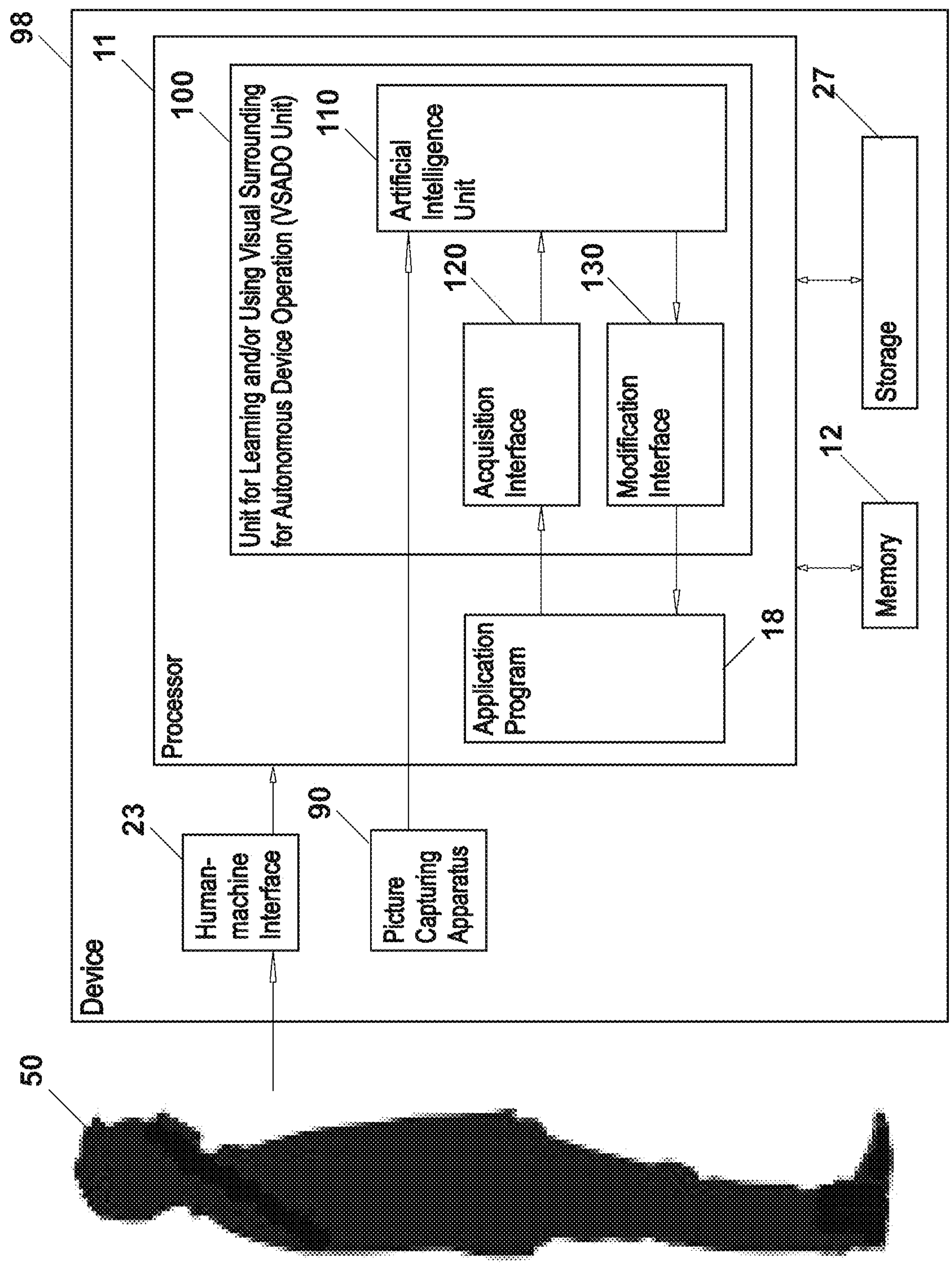

98
Device
11
Processor
100
Unit for Learning and/or Using Visual Surrounding for Autonomous Device Operation (VSADO Unit)
110
Artificial Intelligence Unit
120
Acquisition Interface
130
Modification Interface
Application Program
18
27
Storage
12
Memory
23
Human-machine Interface
90
Picture Capturing Apparatus
50

FIG. 7

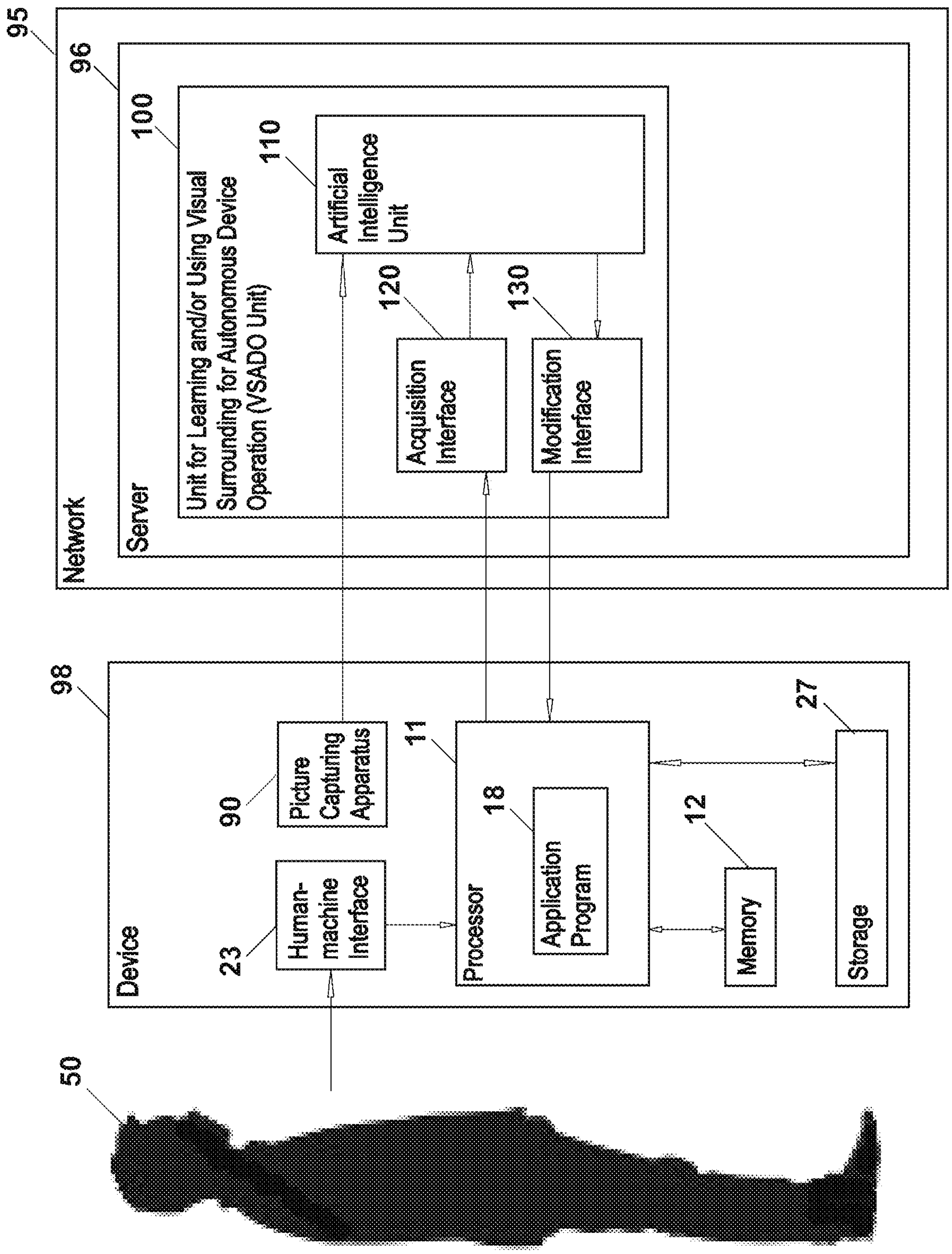

95
Network
96
Server
100
Unit for Learning and/or Using Visual Surrounding for Autonomous Device Operation (VSADO Unit)
110
Artificial Intelligence Unit
120
Acquisition Interface
130
Modification Interface
98
Device
23
Human-machine Interface
90
Picture Capturing Apparatus
11
Processor
18
Application Program
12
Memory
27
Storage
50

FIG. 8

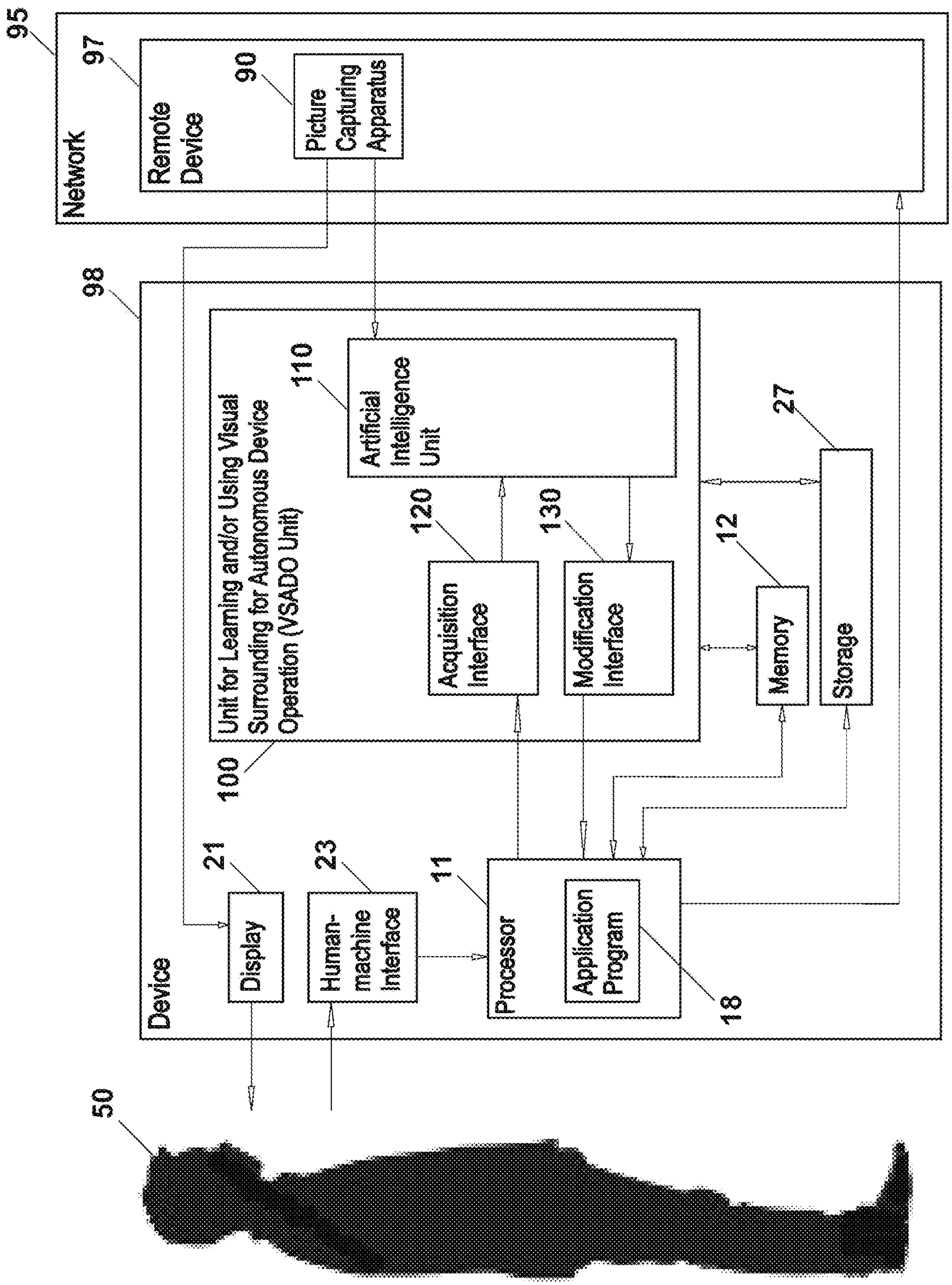

95
Network
97
Remote Device
90
Picture Capturing Apparatus
98
Device
110
Artificial Intelligence Unit
Unit for Learning and/or Using Visual Surrounding for Autonomous Device Operation (VSADO Unit)
100
120
Acquisition Interface
130
Modification Interface
27
Storage
12
Memory
21
Display
23
Human-machine Interface
11
Processor
Application Program
18
50

FIG. 9

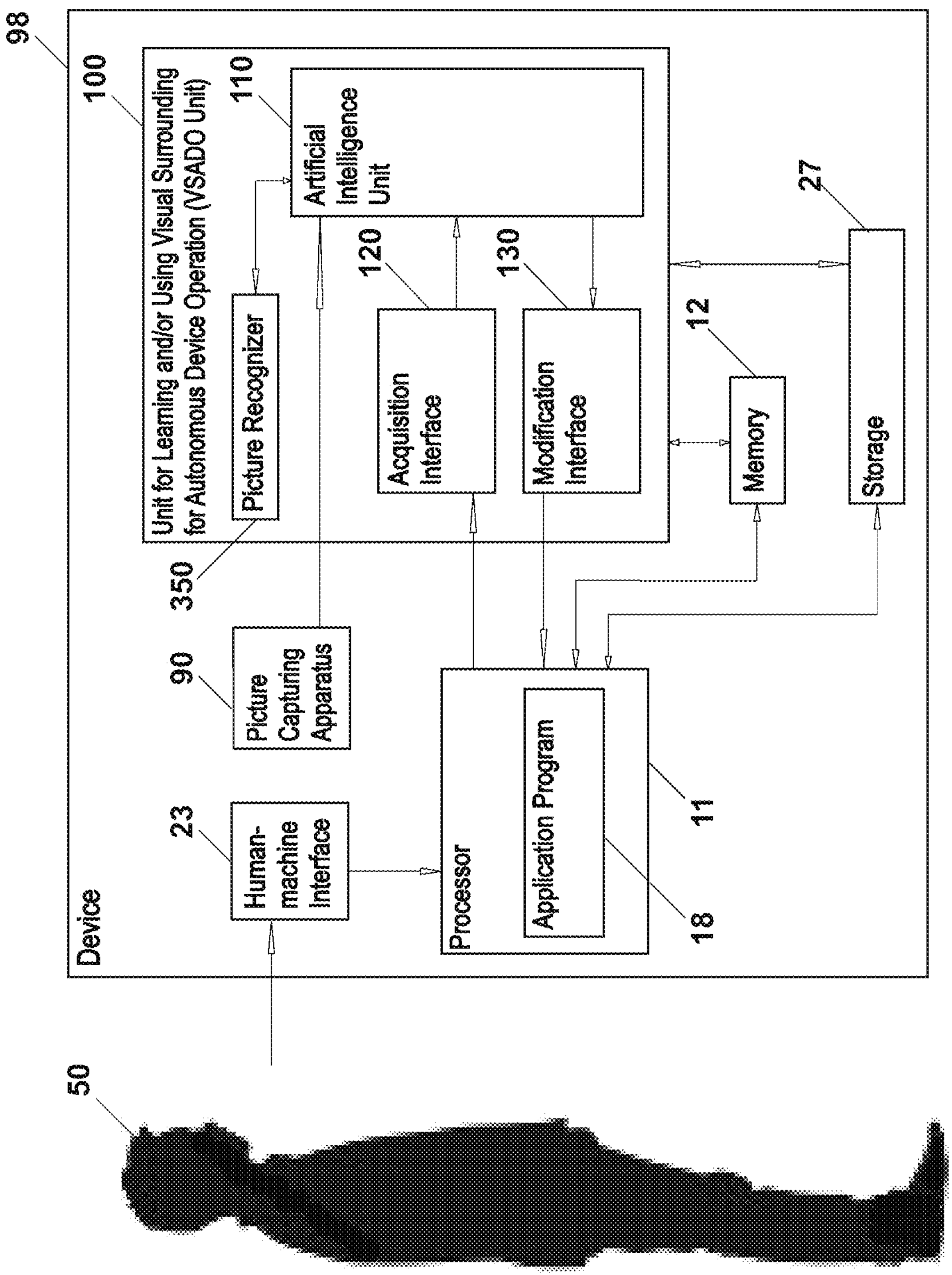
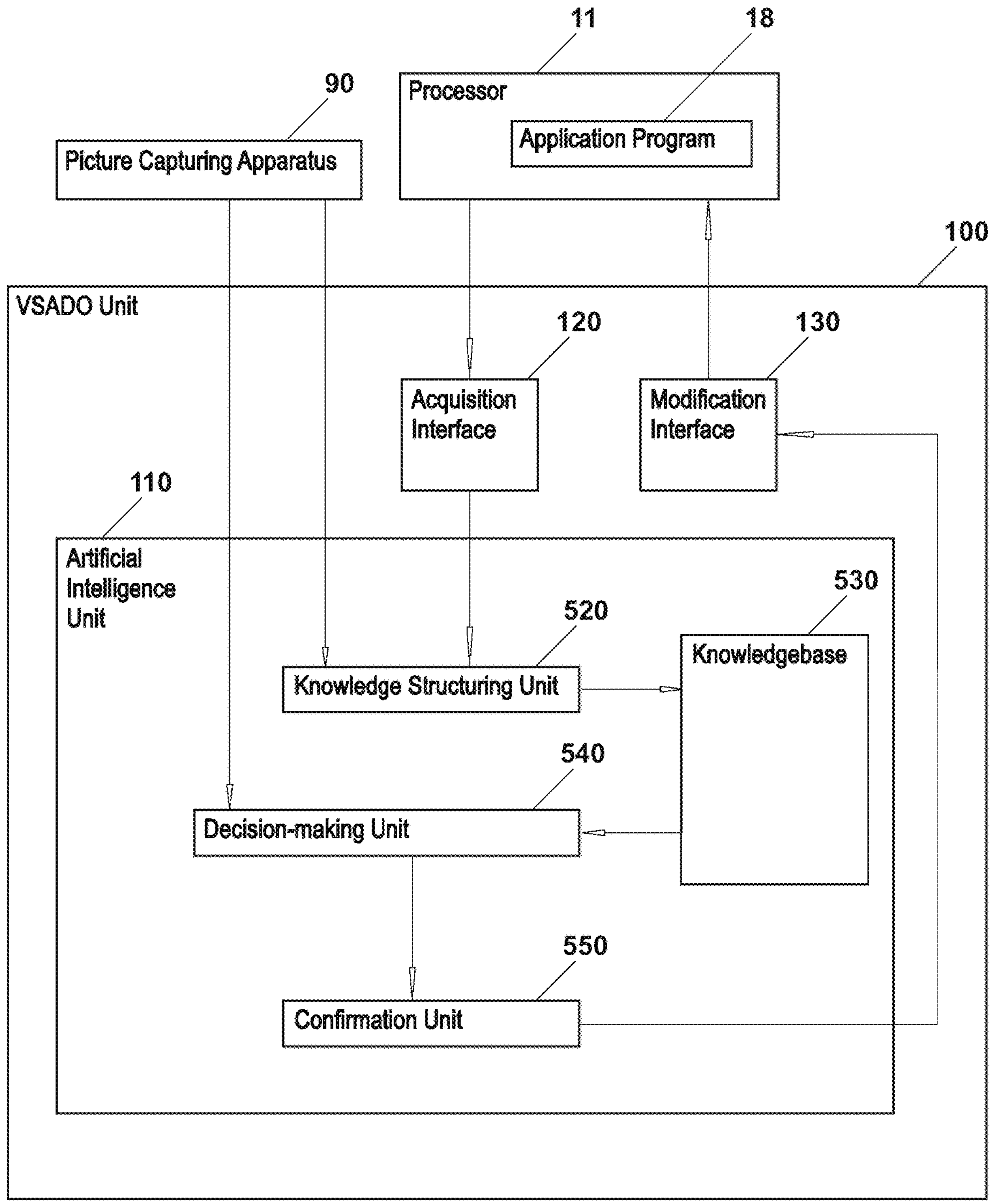

98
Device
100
Unit for Learning and/or Using Visual Surrounding for Autonomous Device Operation (VSADO Unit)
110
Artificial Intelligence Unit
350
Picture Recognizer
120
Acquisition Interface
130
Modification Interface
12
Memory
27
Storage
90
Picture Capturing Apparatus
23
Human-machine Interface
Processor
Application Program
11
18
50

FIG. 10



11
18
90
Processor
Application Program
Picture Capturing Apparatus
100
VSADO Unit
120
130
Acquisition Interface
Modification Interface
110
Artificial Intelligence Unit
530
520
Knowledgebase
Knowledge Structuring Unit
540
Decision-making Unit
550
Confirmation Unit

FIG. 11

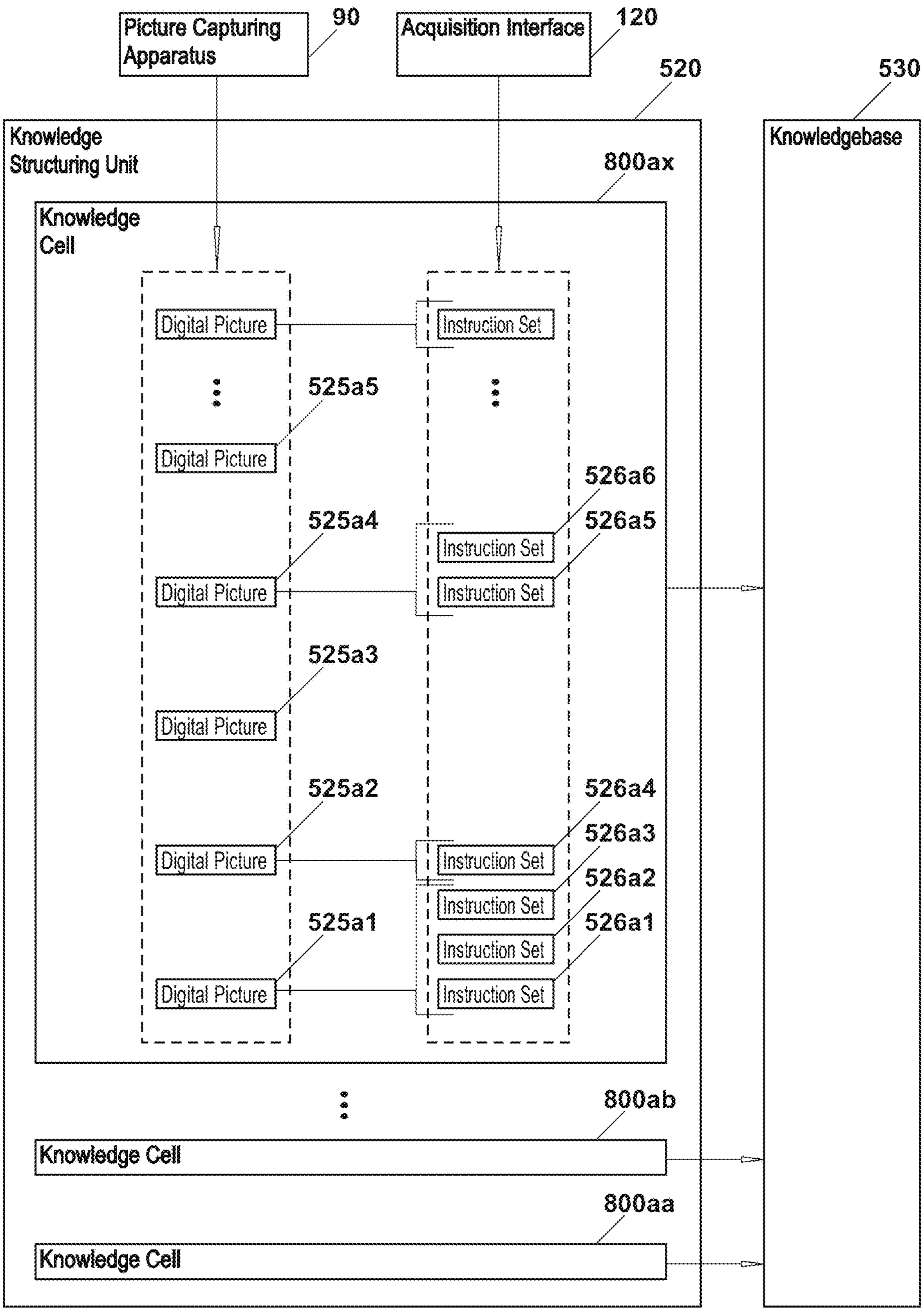
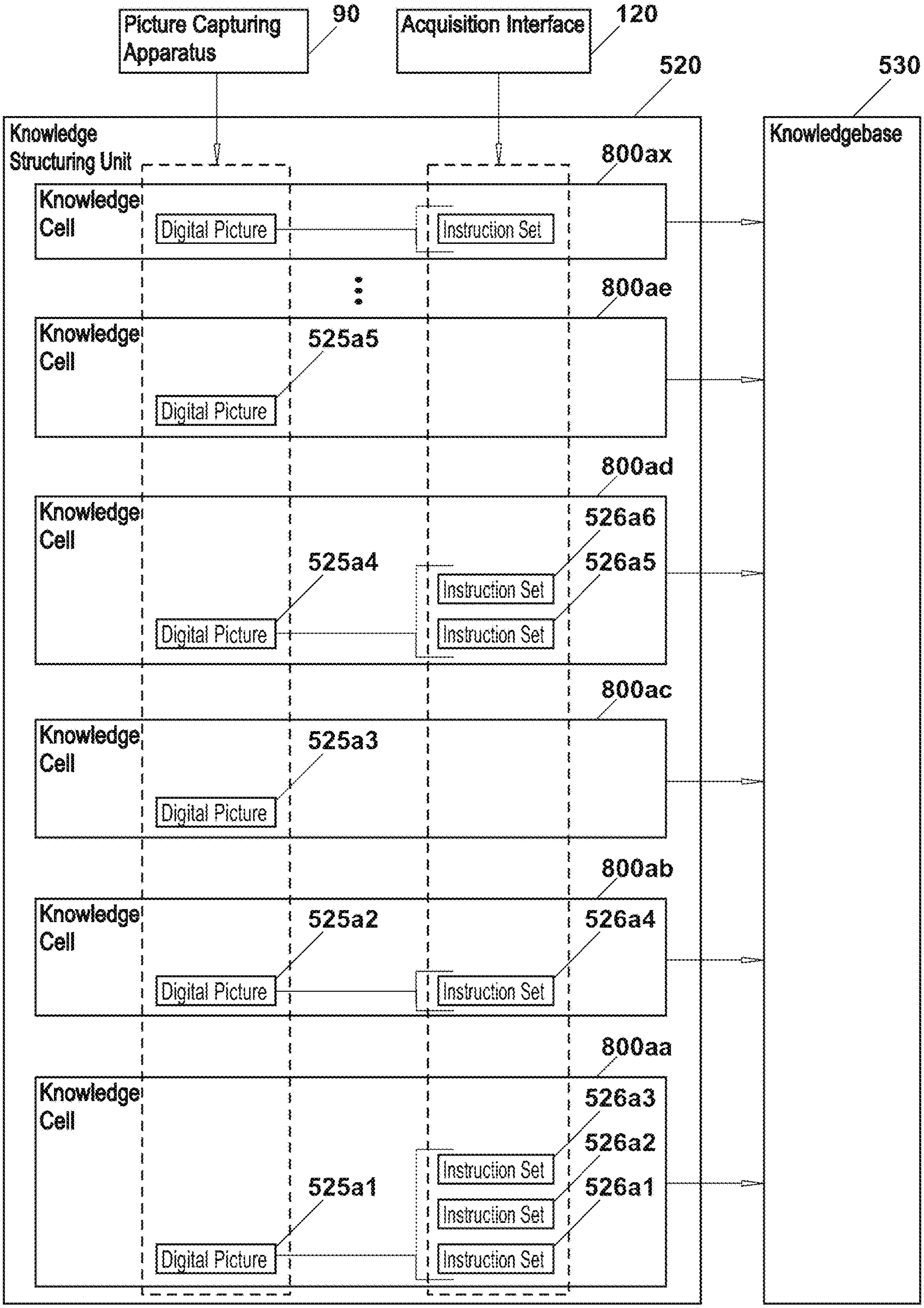

Picture Capturing Apparatus
90
Acquisition Interface
120
520
530
Knowledge Structuring Unit
Knowledgebase
800ax
Knowledge Cell
Digital Picture
Instruction Set
525a5
Digital Picture
526a6
525a4
526a5
Instruction Set
Digital Picture
Instruction Set
525a3
Digital Picture
525a2
526a4
526a3
Digital Picture
Instruction Set
526a2
Instruction Set
525a1
Instruction Set
526a1
Digital Picture
Instruction Set
800ab
Knowledge Cell
800aa
Knowledge Cell

FIG. 12



Picture Capturing Apparatus
90
Acquisition Interface
120
520
530
Knowledge Structuring Unit
Knowledgebase
800ax
Knowledge Cell
Digital Picture
Instruction Set
800ae
Knowledge Cell
525a5
Digital Picture
800ad
Knowledge Cell
526a6
525a4
526a5
Instruction Set
Digital Picture
Instruction Set
800ac
Knowledge Cell
525a3
Digital Picture
800ab
Knowledge Cell
525a2
526a4
Digital Picture
Instruction Set
800aa
Knowledge Cell
526a3
526a2
525a1
Instruction Set
526a1
Instruction Set
Digital Picture
Instruction Set

FIG. 13

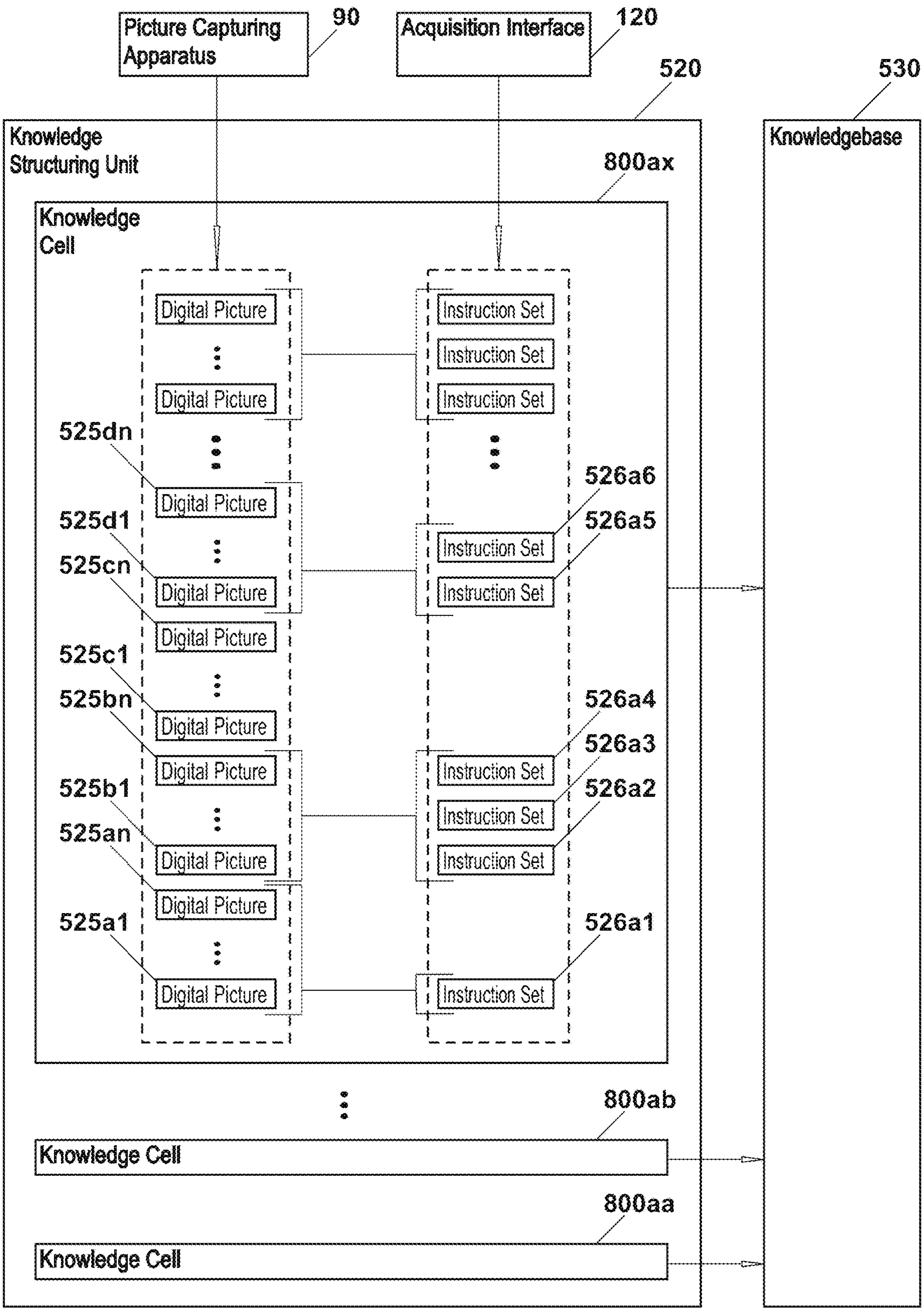
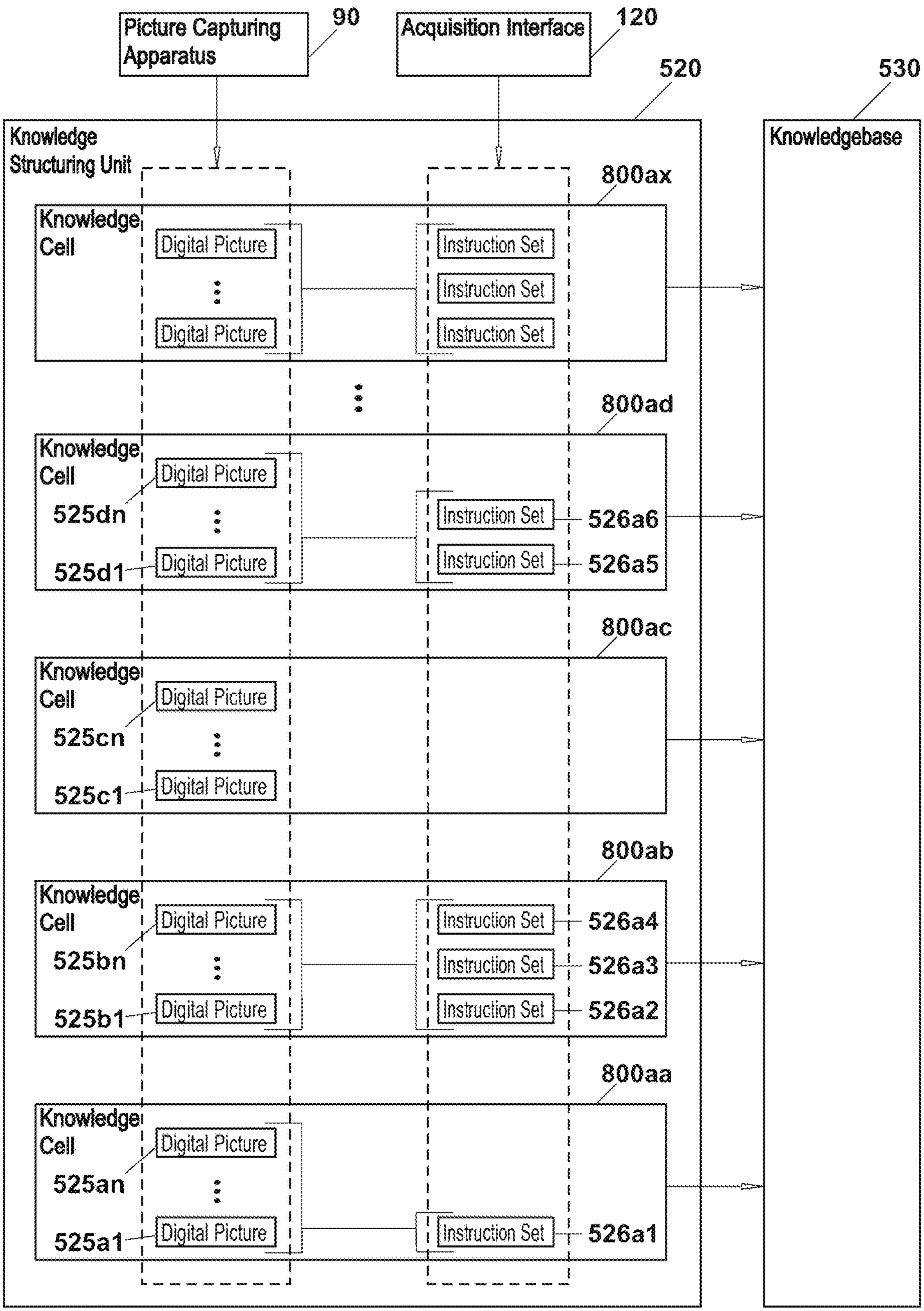

Picture Capturing Apparatus
90
Acquisition Interface
120
520
530
Knowledge Structuring Unit
Knowledgebase
800ax
Knowledge Cell
Digital Picture
Instruction Set
525dn
525d1
525cn
525c1
525bn
525b1
525an
525a1
526a6
526a5
526a4
526a3
526a2
526a1
800ab
Knowledge Cell
800aa
Knowledge Cell

FIG. 14



Picture Capturing Apparatus
90
Acquisition Interface
120
520
530
Knowledge Structuring Unit
Knowledgebase
800ax
Knowledge Cell
Digital Picture
Digital Picture
Instruction Set
Instruction Set
Instruction Set
800ad
Knowledge Cell
Digital Picture
525dn
525d1
Digital Picture
Instruction Set
526a6
Instruction Set
526a5
800ac
Knowledge Cell
Digital Picture
525cn
525c1
Digital Picture
800ab
Knowledge Cell
Digital Picture
525bn
525b1
Digital Picture
Instruction Set
526a4
Instruction Set
526a3
Instruction Set
526a2
800aa
Knowledge Cell
Digital Picture
525an
525a1
Digital Picture
Instruction Set
526a1

FIG. 15

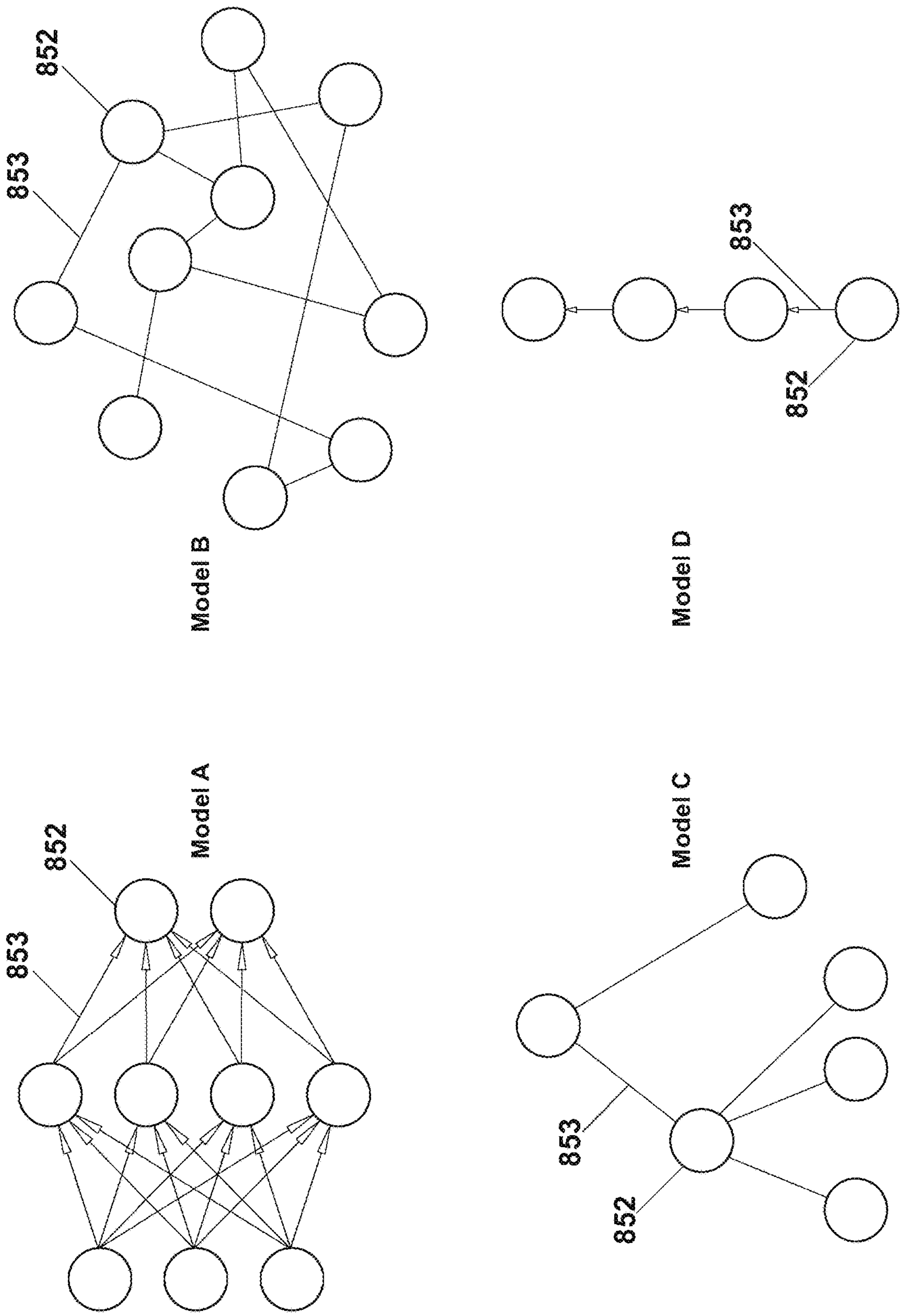


FIG. 16

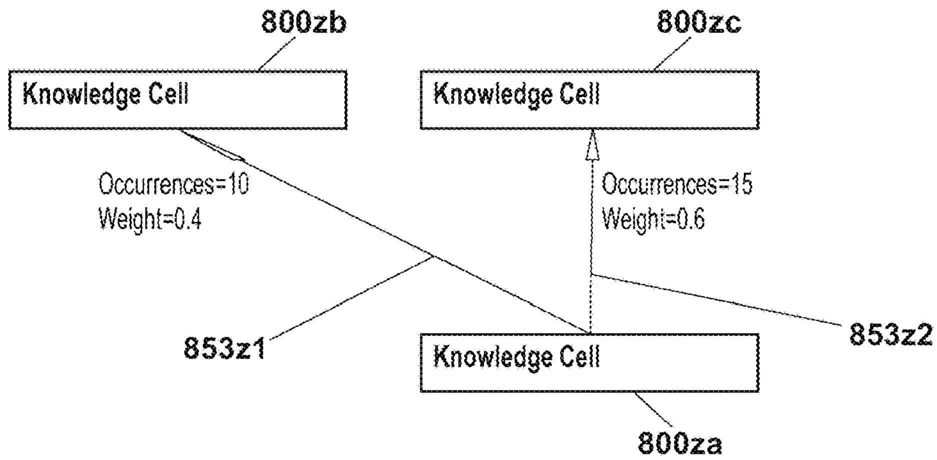


FIG. 17A

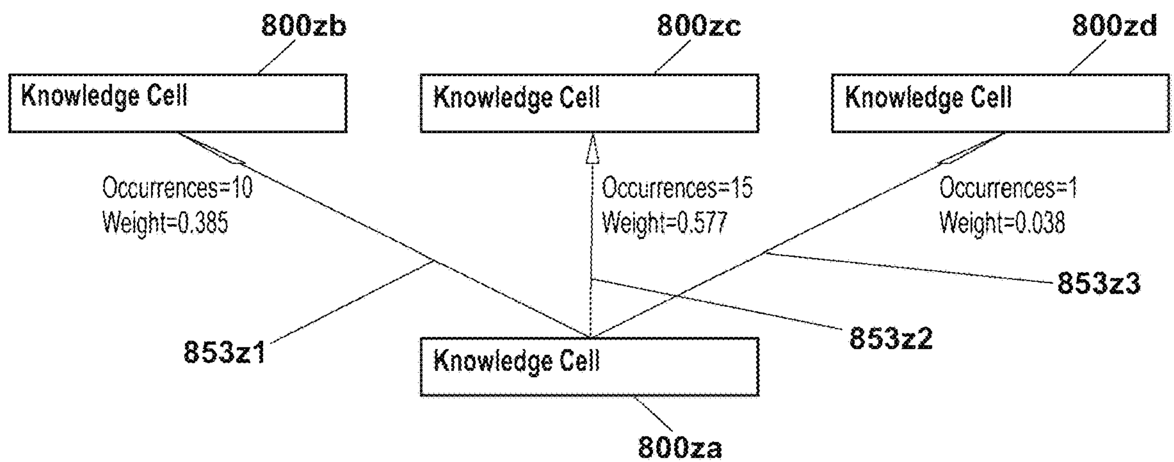


FIG. 17B

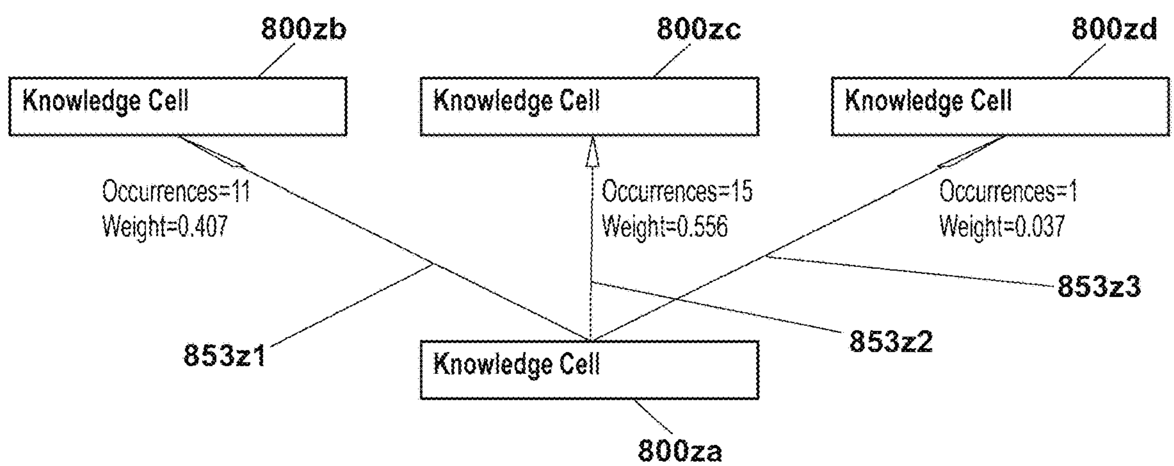
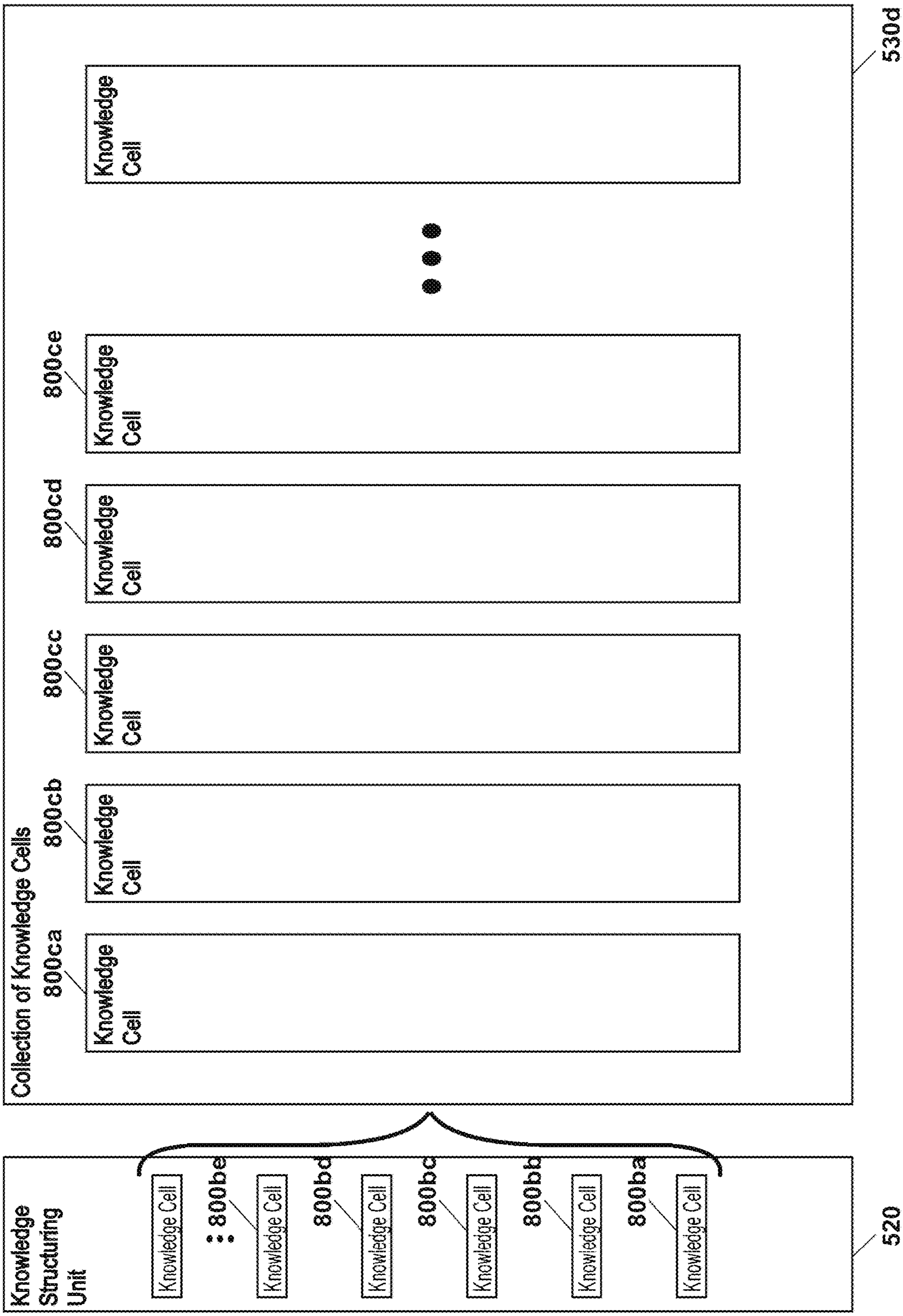
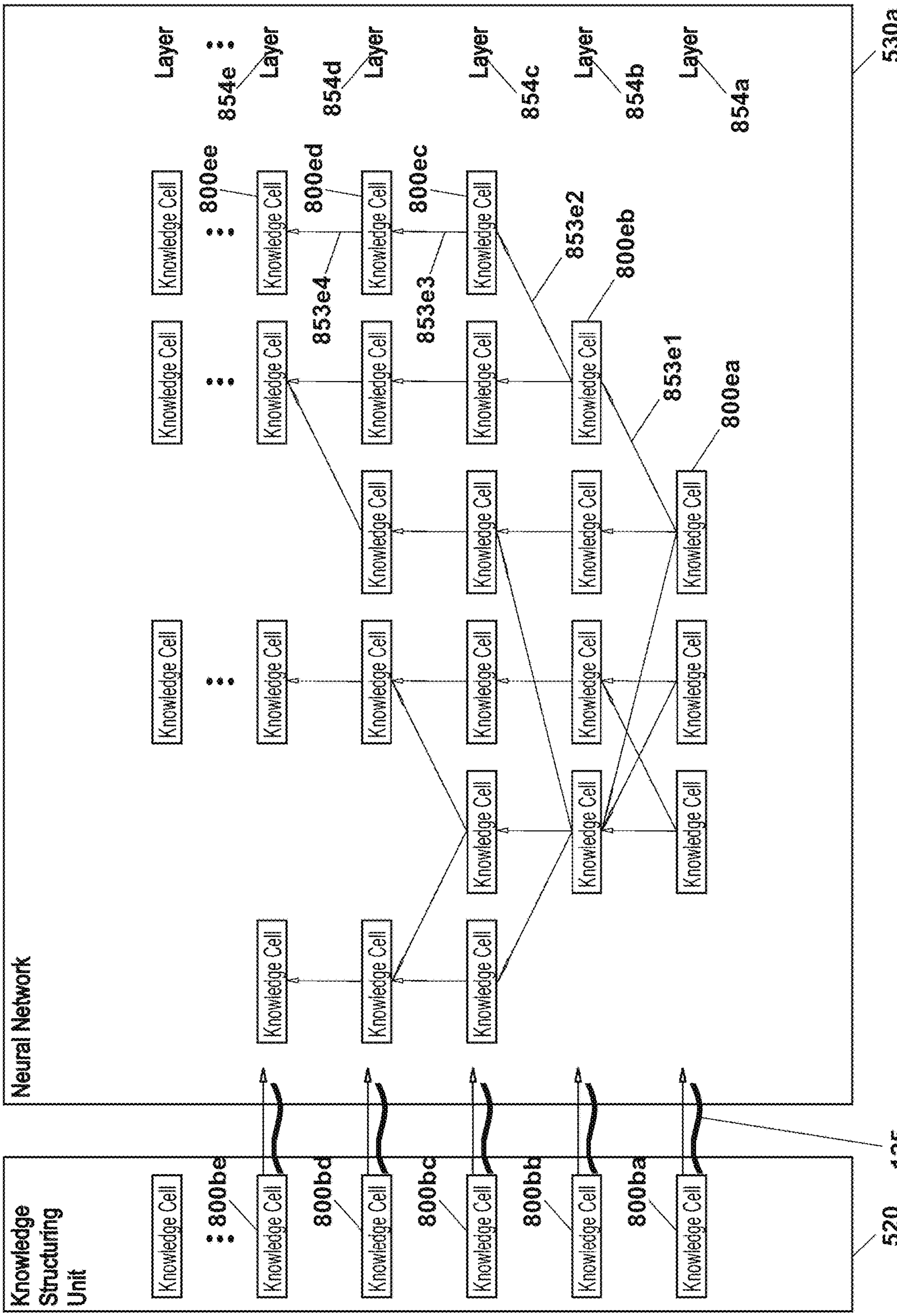


FIG. 17C



Collection of Knowledge Cells
800ca
800cb
800cc
800cd
800ce
Knowledge Cell
Knowledge Cell
Knowledge Cell
Knowledge Cell
Knowledge Cell
Knowledge Cell
530d
Knowledge Structuring Unit
Knowledge Cell
800be
Knowledge Cell
800bd
Knowledge Cell
800bc
Knowledge Cell
800bb
Knowledge Cell
800ba
Knowledge Cell
520

FIG. 18



Knowledge Structuring Unit
Neural Network
Knowledge Cell
Layer
800ba
800bb
800bc
800bd
800be
800ea
800eb
800ec
800ed
800ee
853e1
853e2
853e3
853e4
854a
854b
854c
854d
854e
125
520
530a

FIG. 19

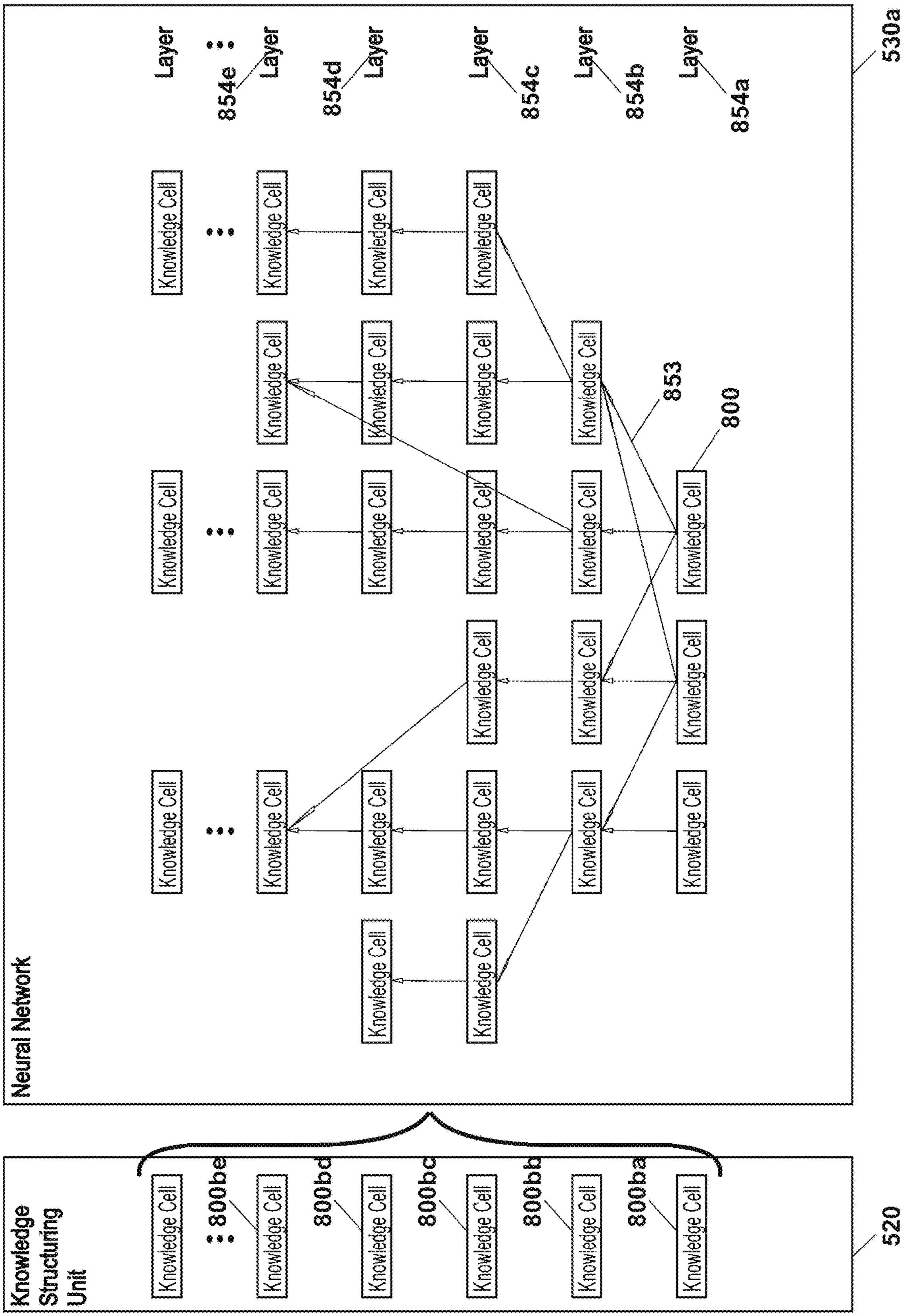

Neural Network
Knowledge Structuring Unit
Knowledge Cell
Layer
854a
854b
854c
854d
854e
853
800
800ba
800bb
800bc
800bd
800be
530a
520

FIG. 20

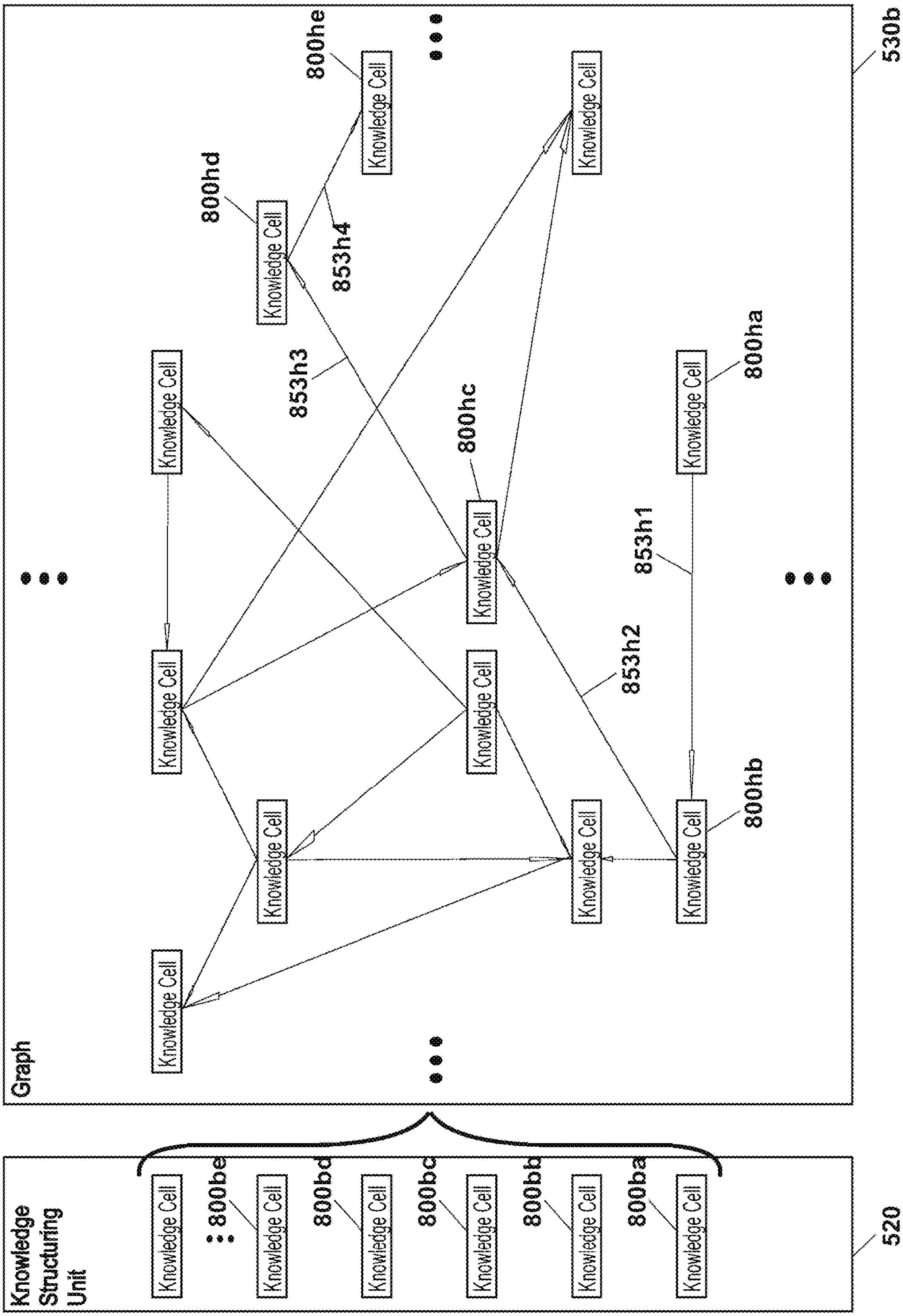

Knowledge Structuring Unit
Knowledge Cell
800be
800bd
800bc
800bb
800ba
520
Graph
Knowledge Cell
800ha
800hb
800hc
800hd
800he
853h1
853h2
853h3
853h4
530b

FIG. 21

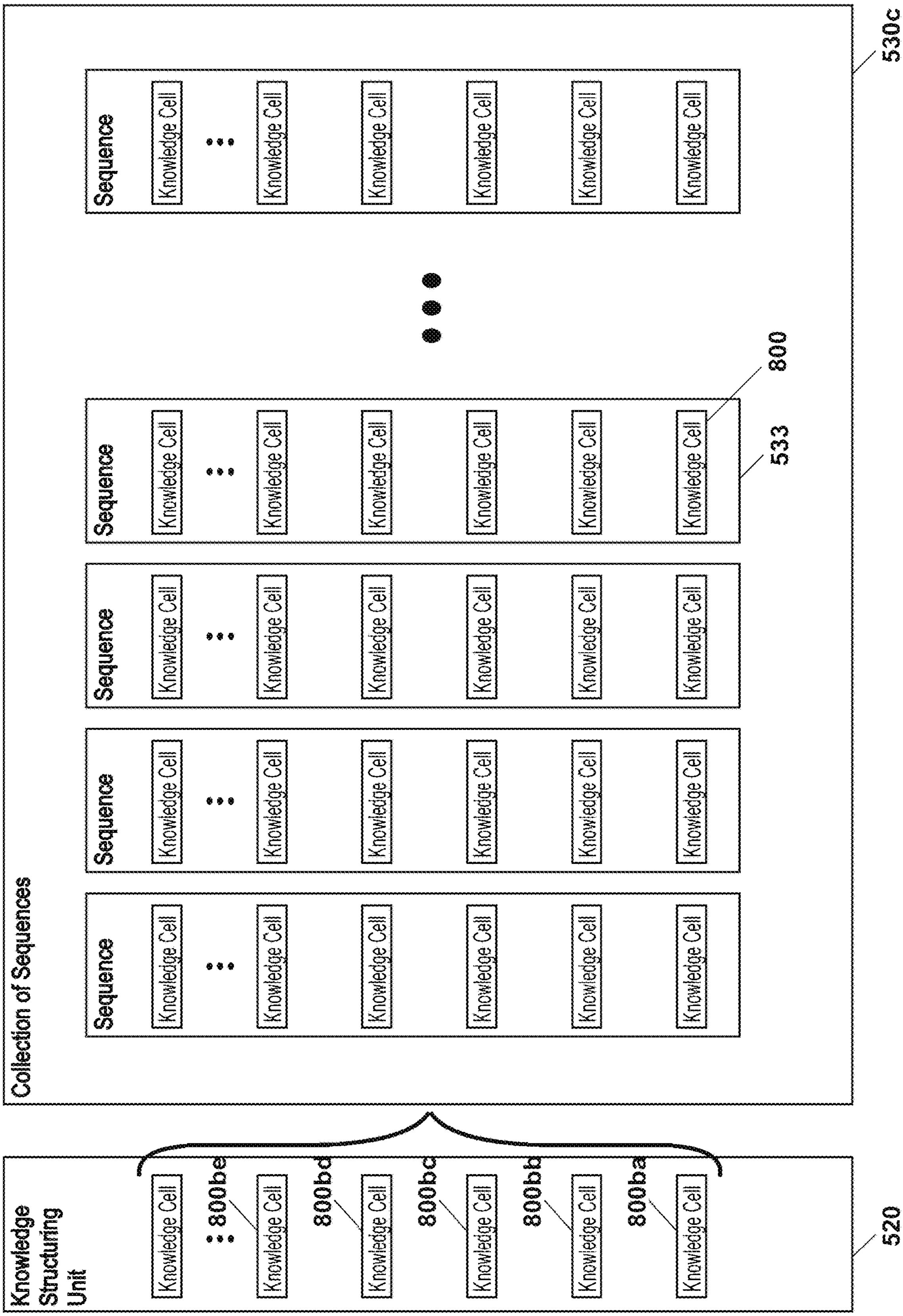
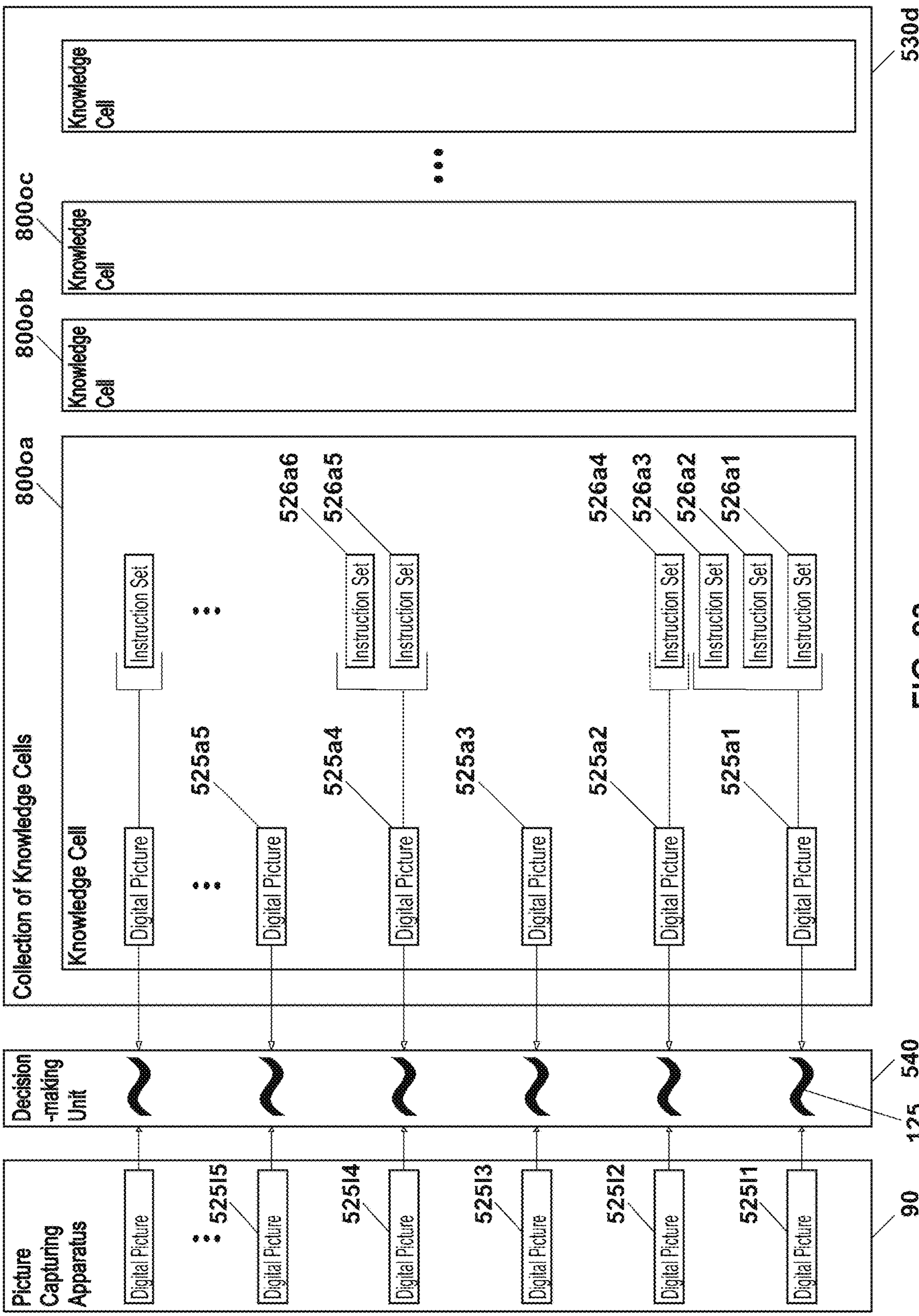

Knowledge Structuring Unit
Knowledge Cell
800be
800bd
800bc
800bb
800ba
520
Collection of Sequences
Sequence
Knowledge Cell
533
800
530c

FIG. 22



Picture Capturing Apparatus
Digital Picture
Digital Picture
Digital Picture
Digital Picture
Digital Picture
Digital Picture
525l5
525l4
525l3
525l2
525l1
90
Decision-making Unit
125
540
Collection of Knowledge Cells
Knowledge Cell
Digital Picture
Digital Picture
Digital Picture
Digital Picture
Digital Picture
Digital Picture
525a5
525a4
525a3
525a2
525a1
Instruction Set
Instruction Set
Instruction Set
Instruction Set
Instruction Set
Instruction Set
Instruction Set
526a6
526a5
526a4
526a3
526a2
526a1
800oa
Knowledge Cell
800ob
Knowledge Cell
800oc
Knowledge Cell
530d

FIG. 23

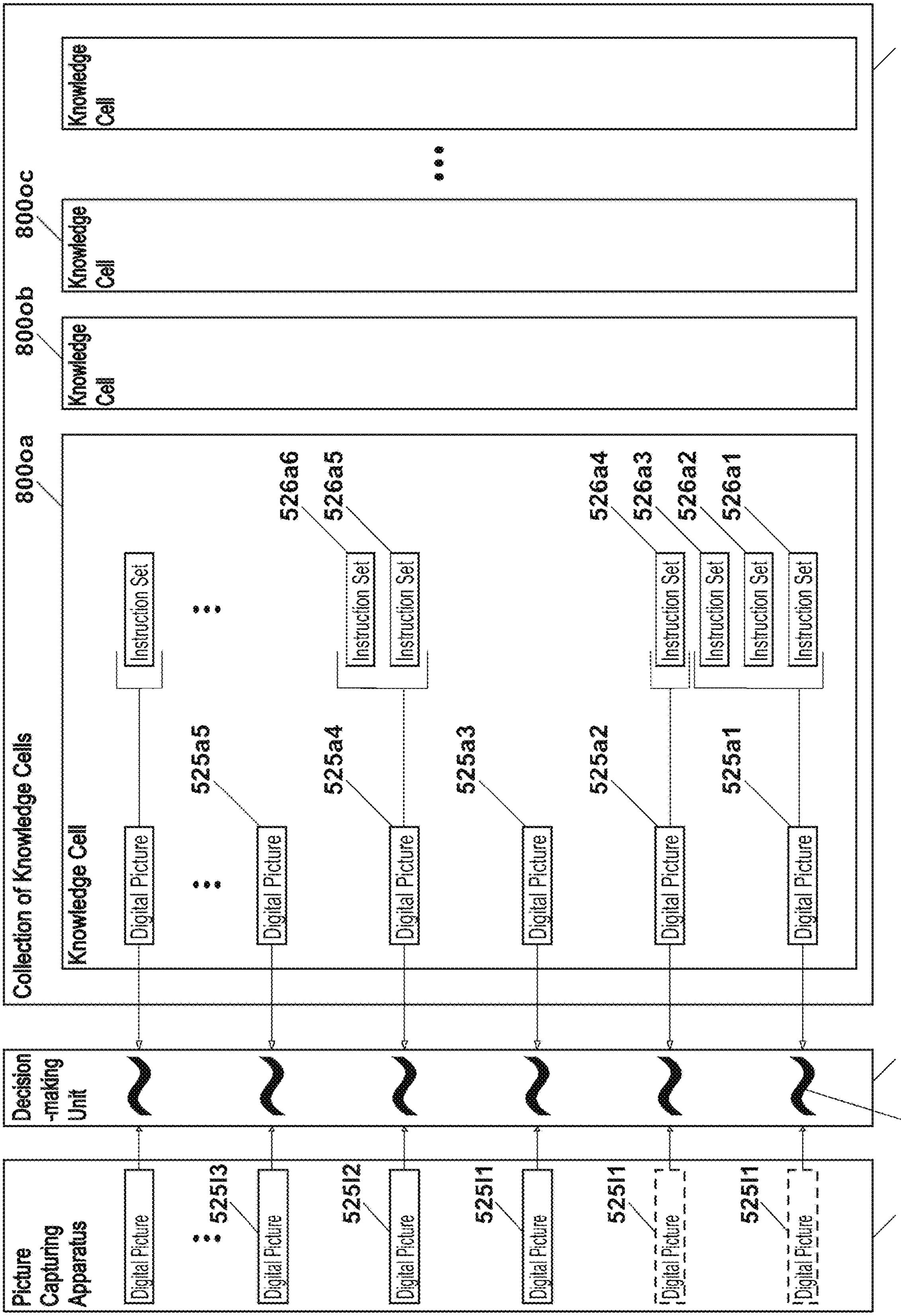

Picture Capturing Apparatus
Digital Picture
525l3
525l2
525l1
90
Decision -making Unit
125
540
Collection of Knowledge Cells
Knowledge Cell
Instruction Set
525a5
525a4
525a3
525a2
525a1
526a6
526a5
526a4
526a3
526a2
526a1
800oa
800ob
800oc
530d

FIG. 24

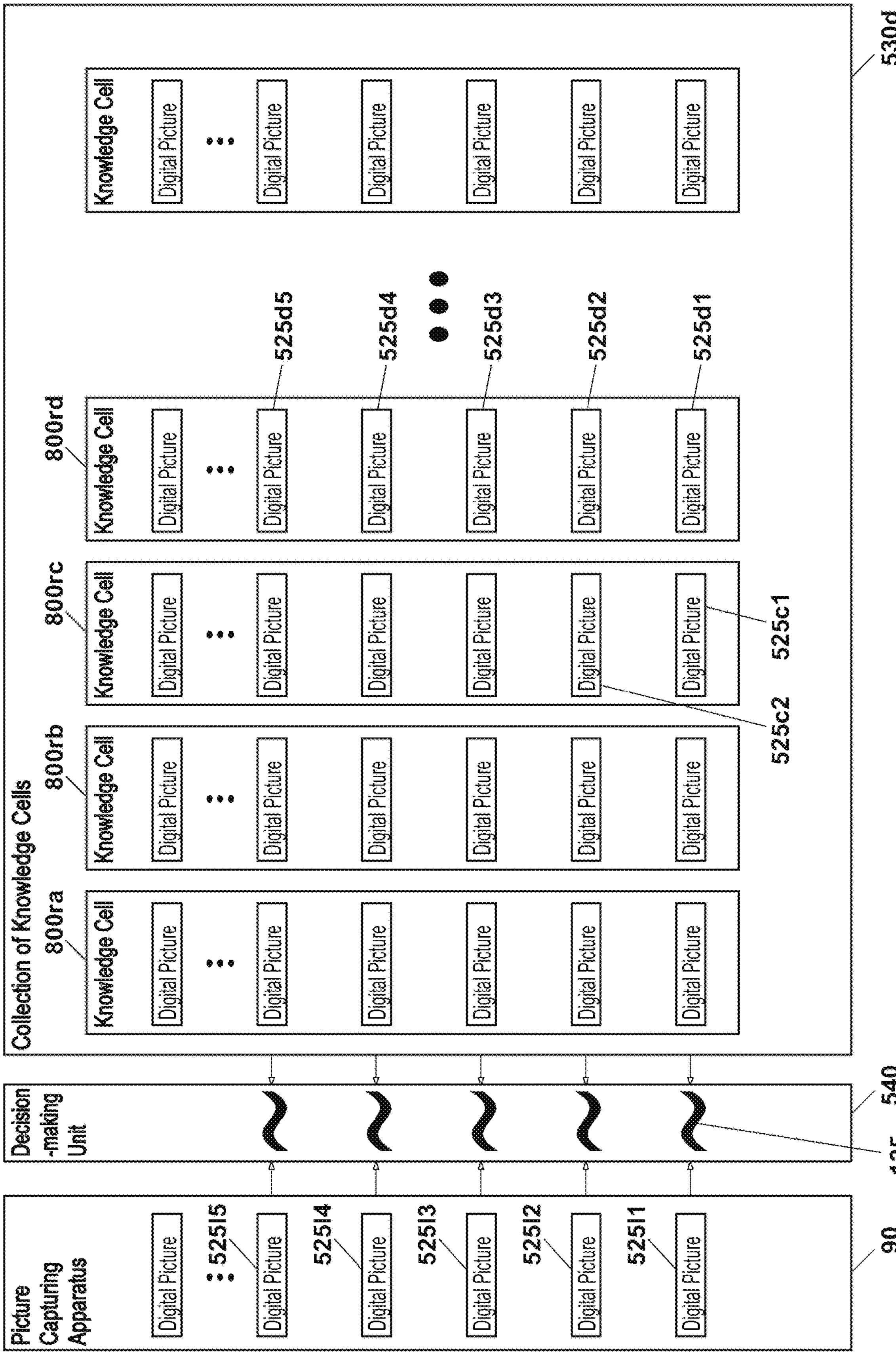

Picture Capturing Apparatus
Digital Picture
525l5
525l4
525l3
525l2
525l1
90
Decision -making Unit
540
125
Collection of Knowledge Cells
Knowledge Cell
800ra
800rb
800rc
800rd
525c1
525c2
525d1
525d2
525d3
525d4
525d5
530d

FIG. 25

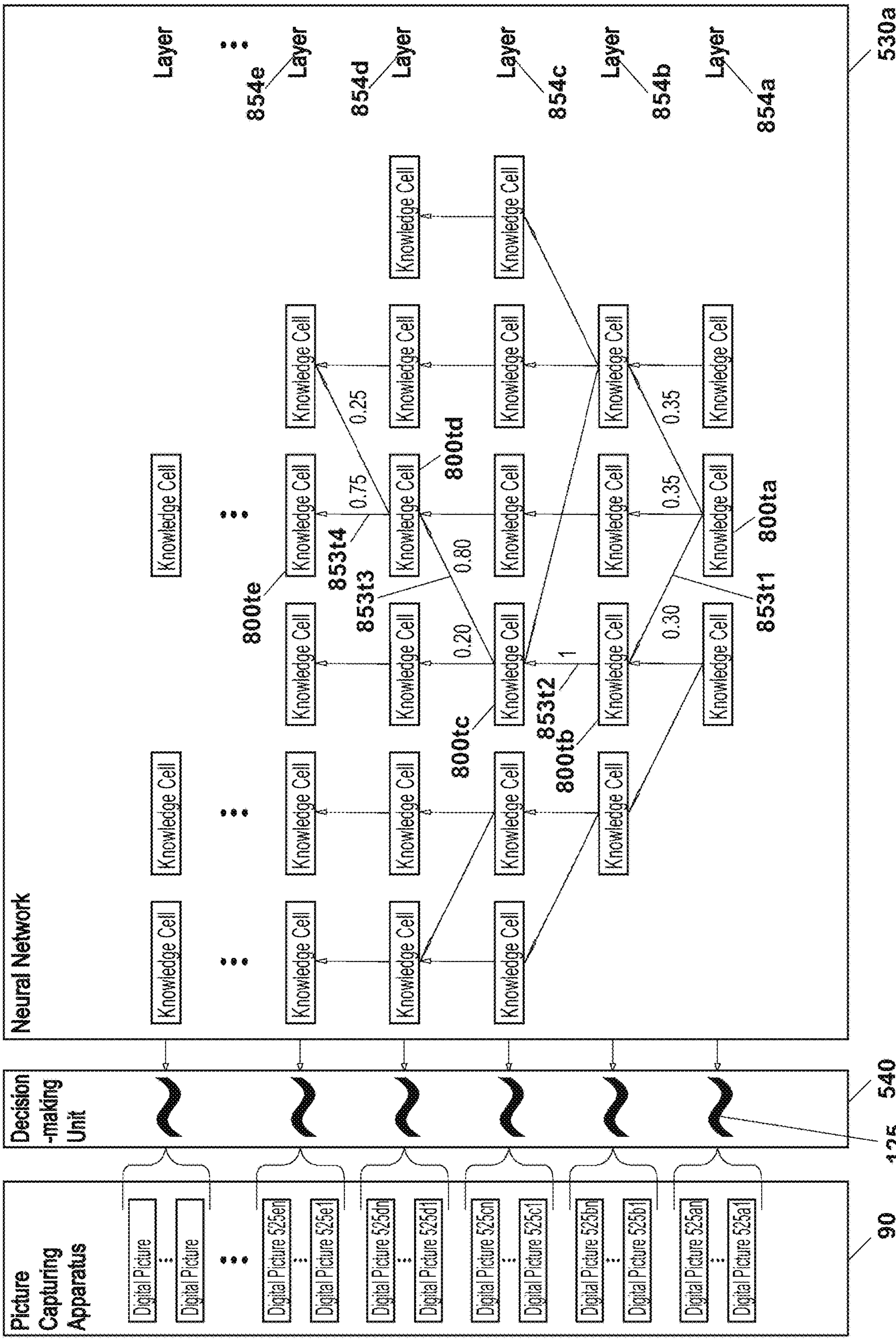

Picture Capturing Apparatus
Digital Picture
Digital Picture 525en
Digital Picture 525e1
Digital Picture 525dn
Digital Picture 525d1
Digital Picture 525cn
Digital Picture 525c1
Digital Picture 525bn
Digital Picture 525b1
Digital Picture 525an
Digital Picture 525a1
90
Decision -making Unit
540
125
Neural Network
Knowledge Cell
Layer
854a
854b
854c
854d
854e
530a
800ta
800tb
800tc
800td
800te
853t1
853t2
853t3
853t4
0.25
0.75
0.80
0.20
0.35
0.35
0.30
1

FIG. 26

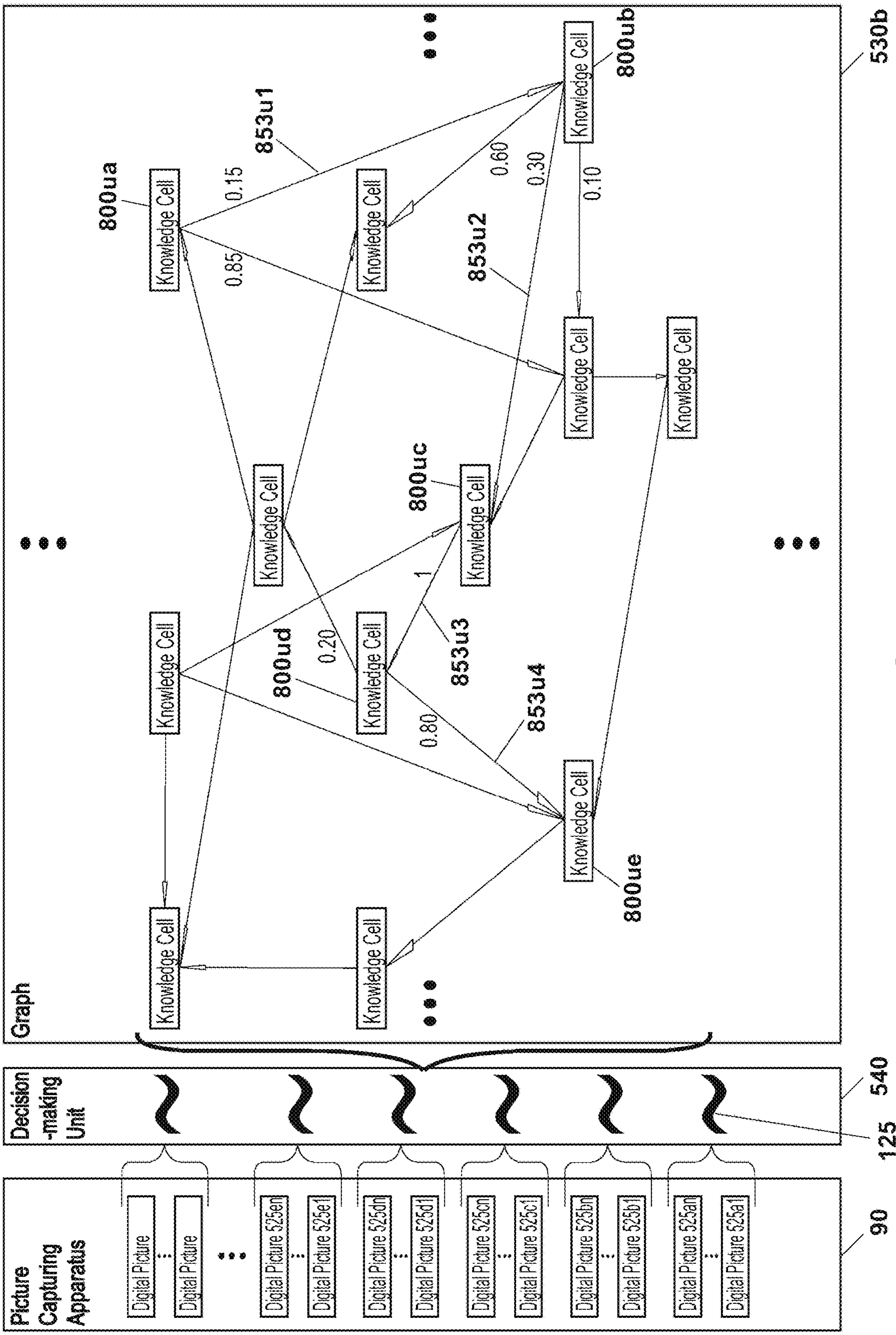

Graph
Knowledge Cell
800ua
800ub
800uc
800ud
800ue
853u1
853u2
853u3
853u4
0.15
0.85
0.60
0.30
0.10
0.20
0.80
1
530b
Decision -making Unit
540
125
Picture Capturing Apparatus
Digital Picture
Digital Picture 525en
Digital Picture 525e1
Digital Picture 525dn
Digital Picture 525d1
Digital Picture 525cn
Digital Picture 525c1
Digital Picture 525bn
Digital Picture 525b1
Digital Picture 525an
Digital Picture 525a1
90

FIG. 27

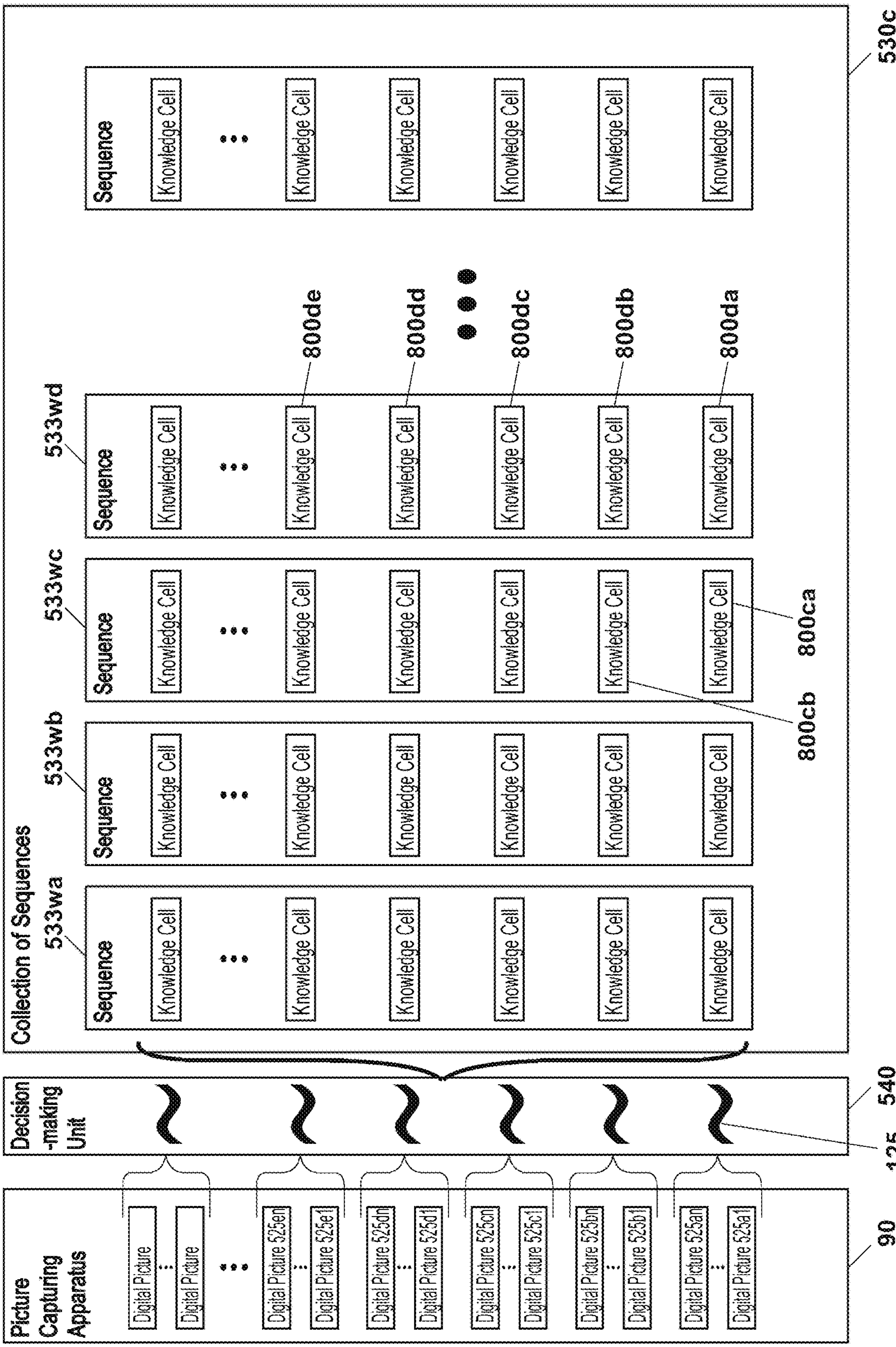

Picture Capturing Apparatus
90
Digital Picture
Digital Picture
Digital Picture 525en
Digital Picture 525e1
Digital Picture 525dn
Digital Picture 525d1
Digital Picture 525cn
Digital Picture 525c1
Digital Picture 525bn
Digital Picture 525b1
Digital Picture 525an
Digital Picture 525a1
Decision -making Unit
540
125
Collection of Sequences
530c
533wa
533wb
533wc
533wd
Sequence
Knowledge Cell
800ca
800cb
800da
800db
800dc
800dd
800de

FIG. 28

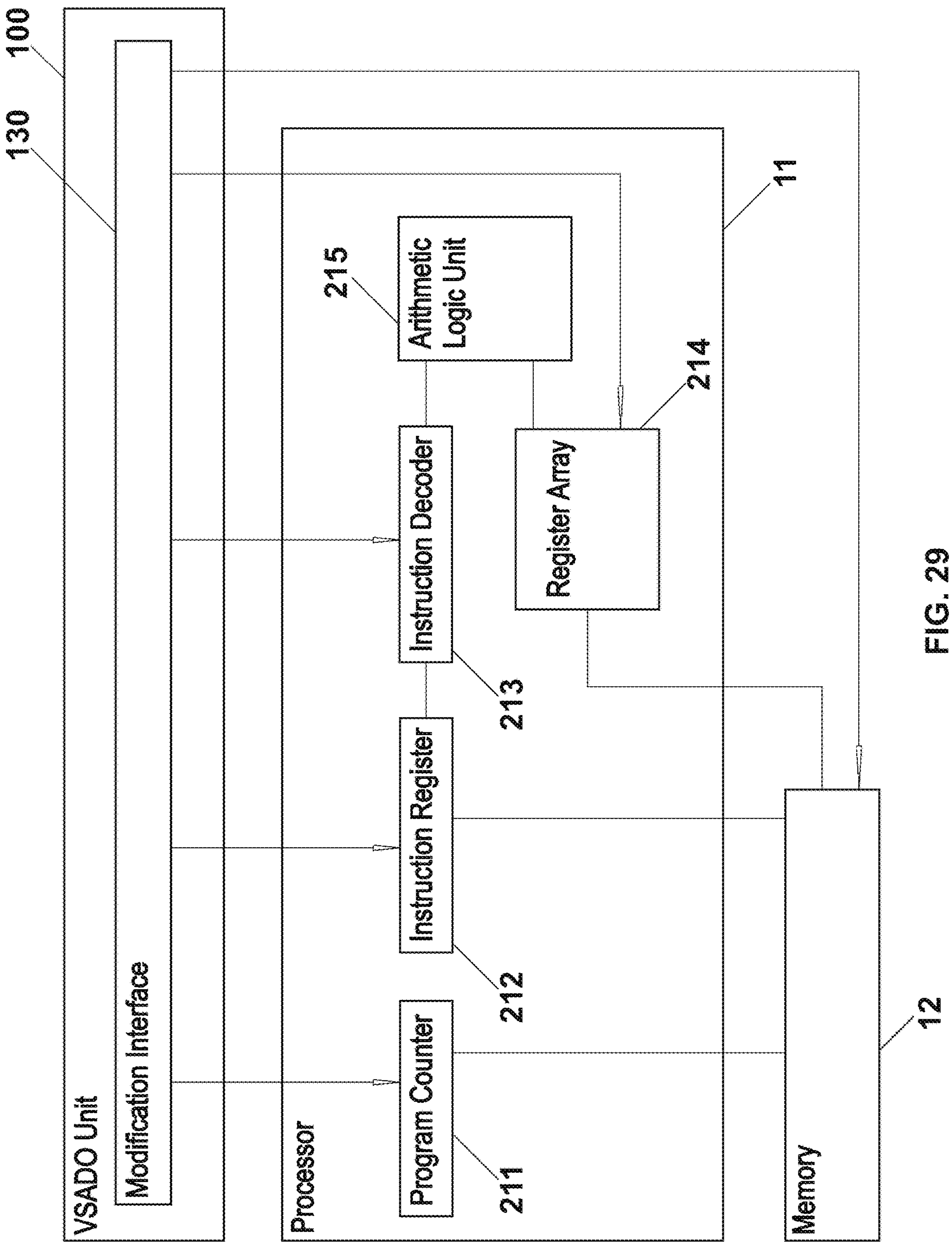

100
130
VSADO Unit
Modification Interface
Processor
11
Program Counter
211
Instruction Register
212
Instruction Decoder
213
Register Array
214
Arithmetic
Logic Unit
215
Memory
12

FIG. 29

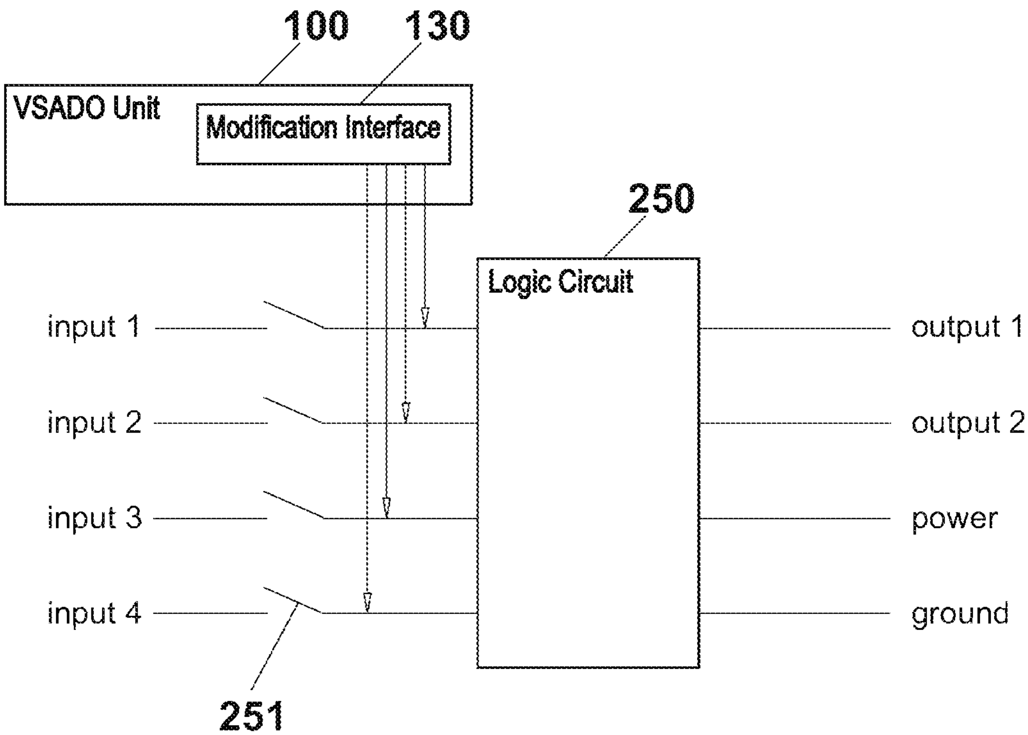


FIG. 30A

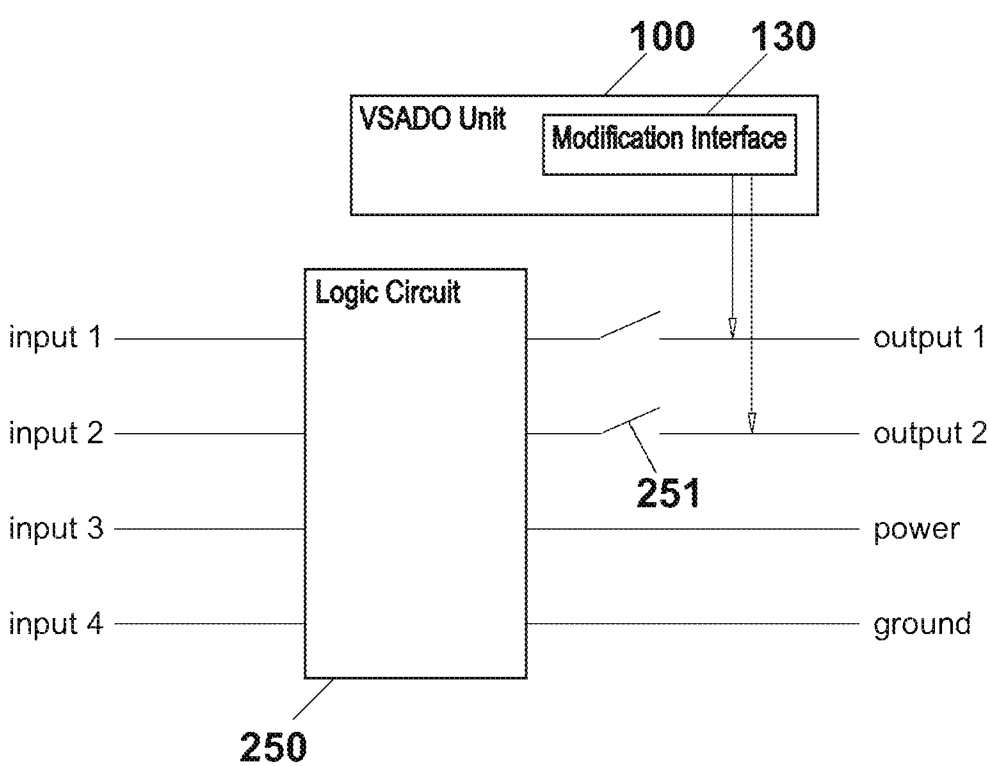


FIG. 30B

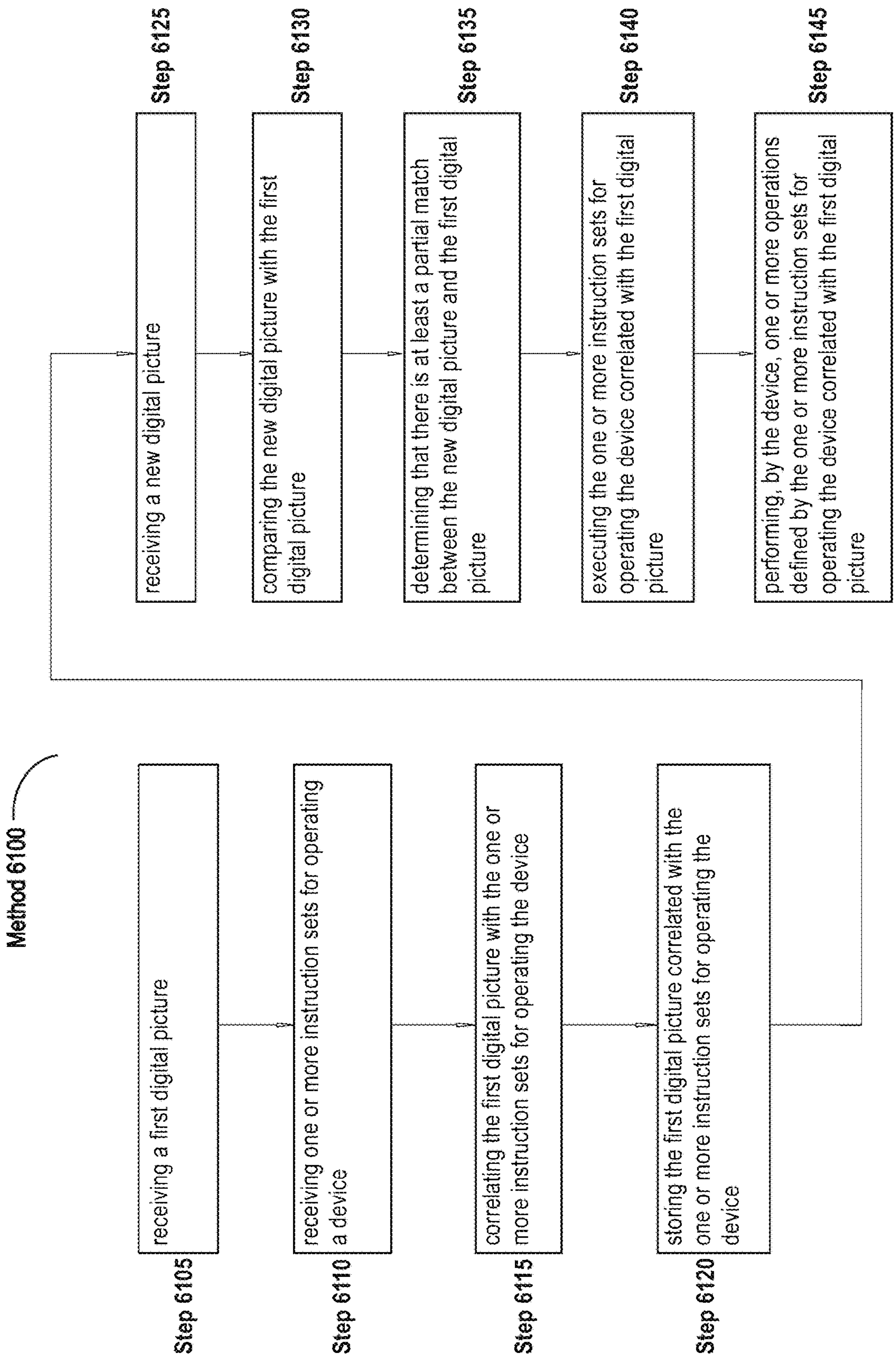

Method 6100
Step 6105
receiving a first digital picture
Step 6110
receiving one or more instruction sets for operating a device
Step 6115
correlating the first digital picture with the one or more instruction sets for operating the device
Step 6120
storing the first digital picture correlated with the one or more instruction sets for operating the device
Step 6125
receiving a new digital picture
Step 6130
comparing the new digital picture with the first digital picture
Step 6135
determining that there is at least a partial match between the new digital picture and the first digital picture
Step 6140
executing the one or more instruction sets for operating the device correlated with the first digital picture
Step 6145
performing, by the device, one or more operations defined by the one or more instruction sets for operating the device correlated with the first digital picture

FIG. 31

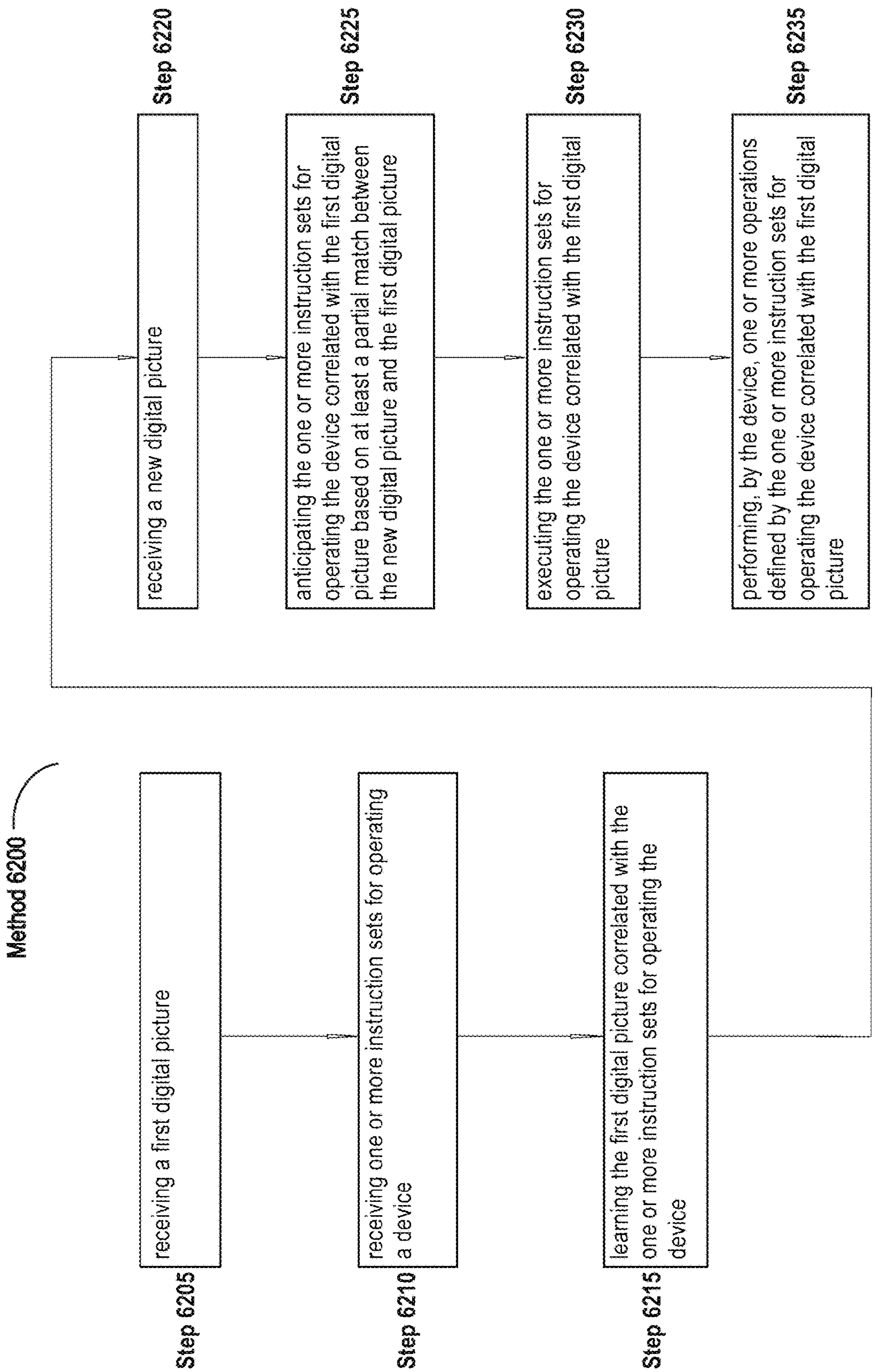

Method 6200
receiving a first digital picture
Step 6205
receiving one or more instruction sets for operating a device
Step 6210
learning the first digital picture correlated with the one or more instruction sets for operating the device
Step 6215
receiving a new digital picture
Step 6220
anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture
Step 6225
executing the one or more instruction sets for operating the device correlated with the first digital picture
Step 6230
performing, by the device, one or more operations defined by the one or more instruction sets for operating the device correlated with the first digital picture
Step 6235

FIG. 32

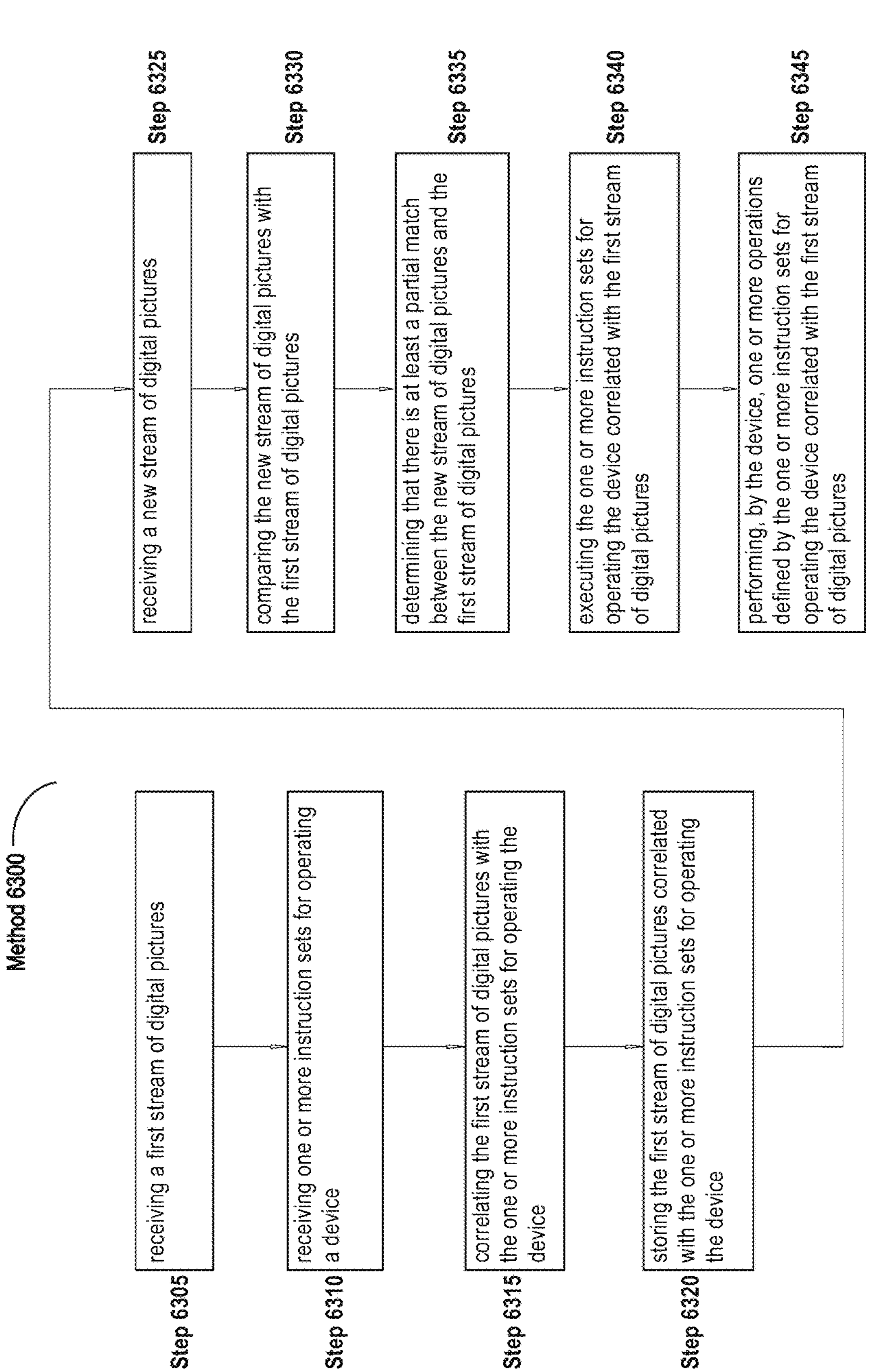

Method 6300
Step 6305
receiving a first stream of digital pictures
Step 6310
receiving one or more instruction sets for operating a device
Step 6315
correlating the first stream of digital pictures with the one or more instruction sets for operating the device
Step 6320
storing the first stream of digital pictures correlated with the one or more instruction sets for operating the device
Step 6325
receiving a new stream of digital pictures
Step 6330
comparing the new stream of digital pictures with the first stream of digital pictures
Step 6335
determining that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures
Step 6340
executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures
Step 6345
performing, by the device, one or more operations defined by the one or more instruction sets for operating the device correlated with the first stream of digital pictures

FIG. 33

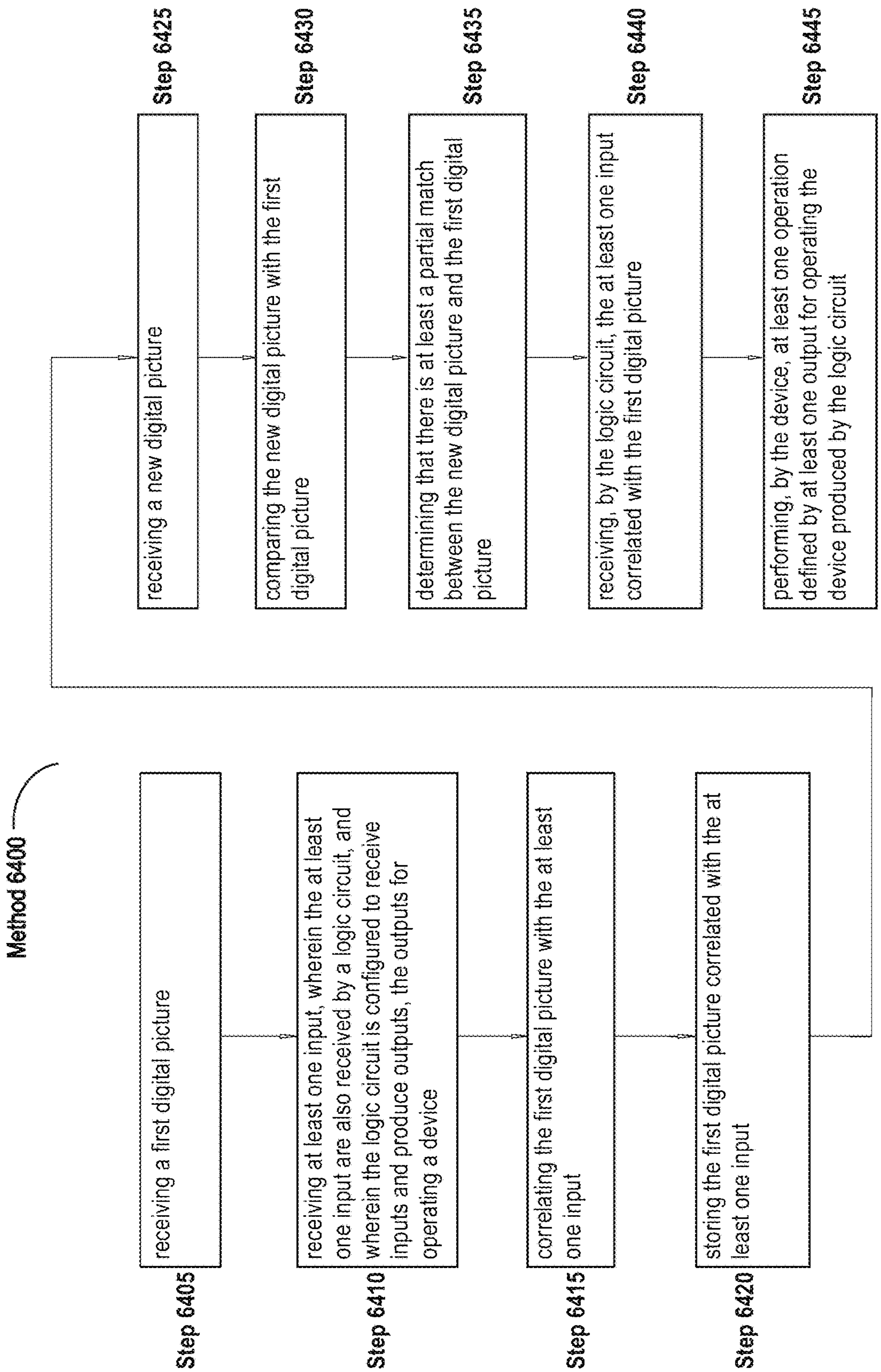

Method 6400
Step 6405
receiving a first digital picture
Step 6410
receiving at least one input, wherein the at least one input are also received by a logic circuit, and wherein the logic circuit is configured to receive inputs and produce outputs, the outputs for operating a device
Step 6415
correlating the first digital picture with the at least one input
Step 6420
storing the first digital picture correlated with the at least one input
Step 6425
receiving a new digital picture
Step 6430
comparing the new digital picture with the first digital picture
Step 6435
determining that there is at least a partial match between the new digital picture and the first digital picture
Step 6440
receiving, by the logic circuit, the at least one input correlated with the first digital picture
Step 6445
performing, by the device, at least one operation defined by at least one output for operating the device produced by the logic circuit

FIG. 34

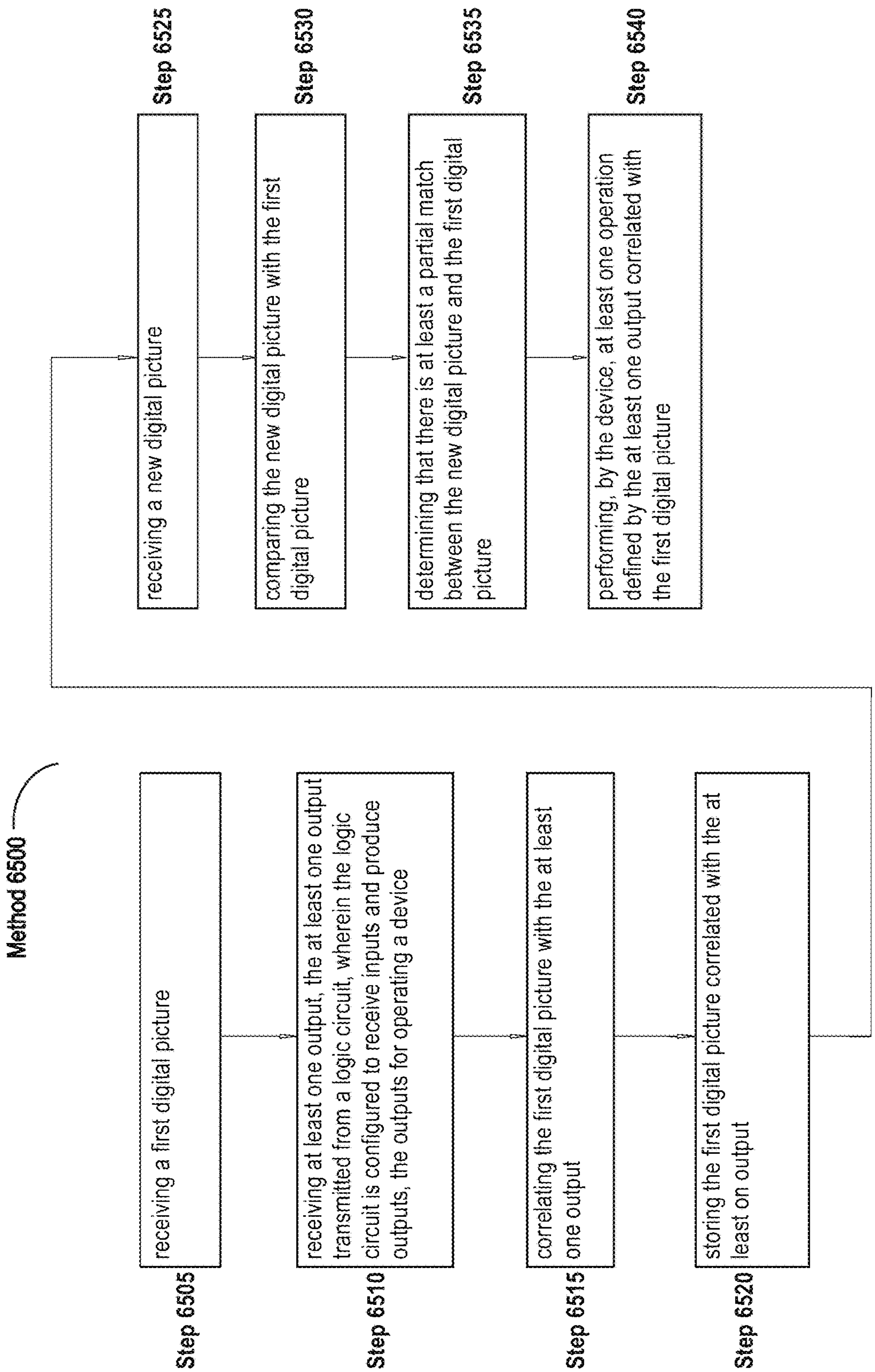

Method 6500
receiving a first digital picture
Step 6505
receiving at least one output, the at least one output transmitted from a logic circuit, wherein the logic circuit is configured to receive inputs and produce outputs, the outputs for operating a device
Step 6510
correlating the first digital picture with the at least one output
Step 6515
storing the first digital picture correlated with the at least on output
Step 6520
receiving a new digital picture
Step 6525
comparing the new digital picture with the first digital picture
Step 6530
determining that there is at least a partial match between the new digital picture and the first digital picture
Step 6535
performing, by the device, at least one operation defined by the at least one output correlated with the first digital picture
Step 6540

FIG. 35

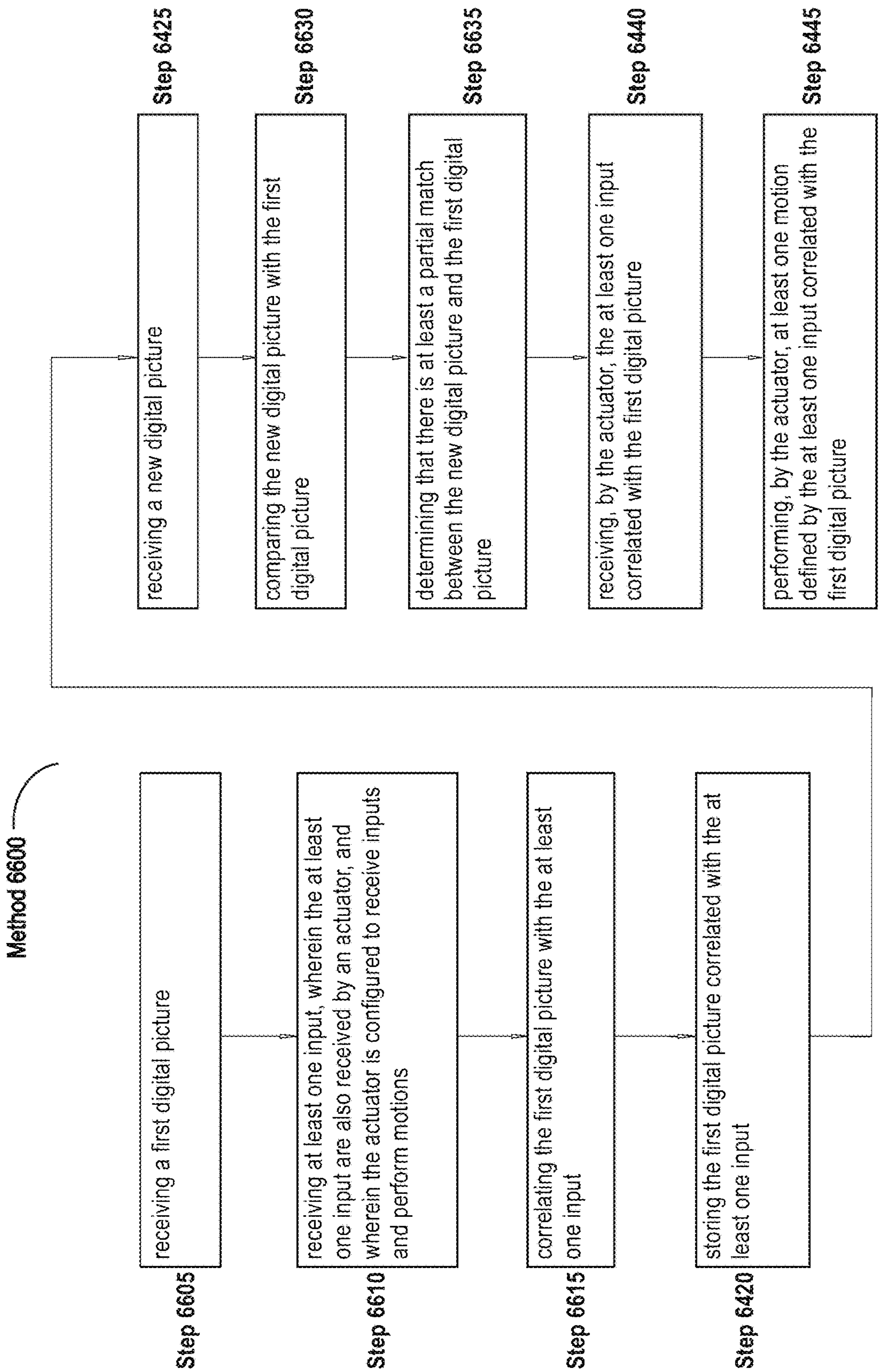

Method 6600
Step 6605
receiving a first digital picture
Step 6610
receiving at least one input, wherein the at least one input are also received by an actuator, and wherein the actuator is configured to receive inputs and perform motions
Step 6615
correlating the first digital picture with the at least one input
Step 6420
storing the first digital picture correlated with the at least one input
Step 6425
receiving a new digital picture
Step 6630
comparing the new digital picture with the first digital picture
Step 6635
determining that there is at least a partial match between the new digital picture and the first digital picture
Step 6440
receiving, by the actuator, the at least one input correlated with the first digital picture
Step 6445
performing, by the actuator, at least one motion defined by the at least one input correlated with the first digital picture

FIG. 36

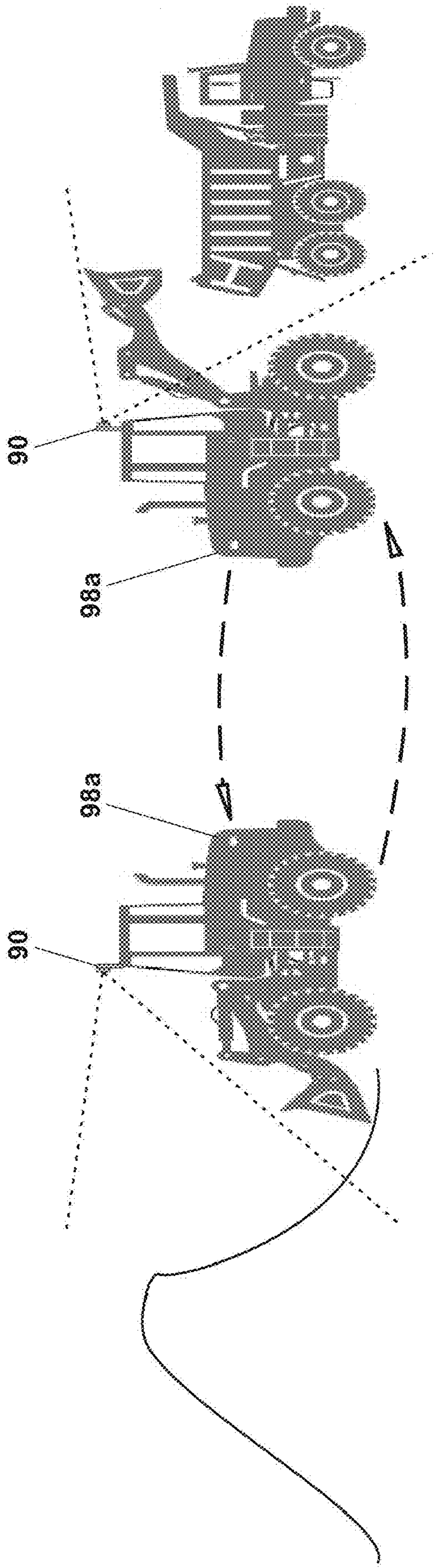


FIG. 37

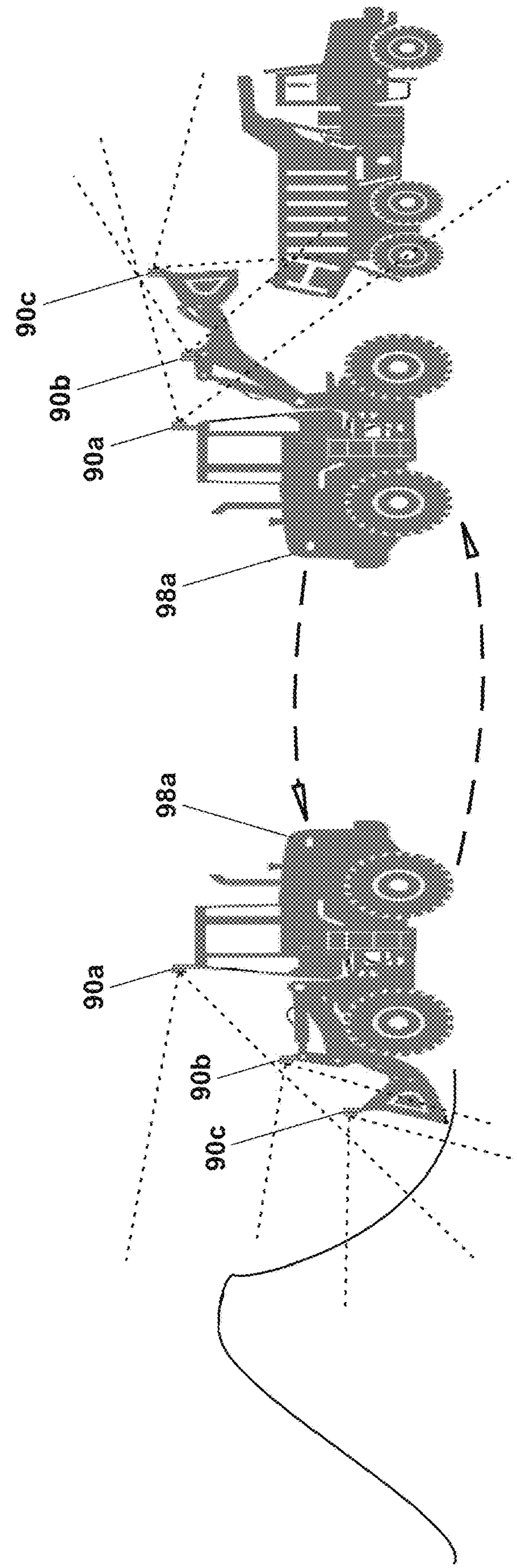


FIG. 38

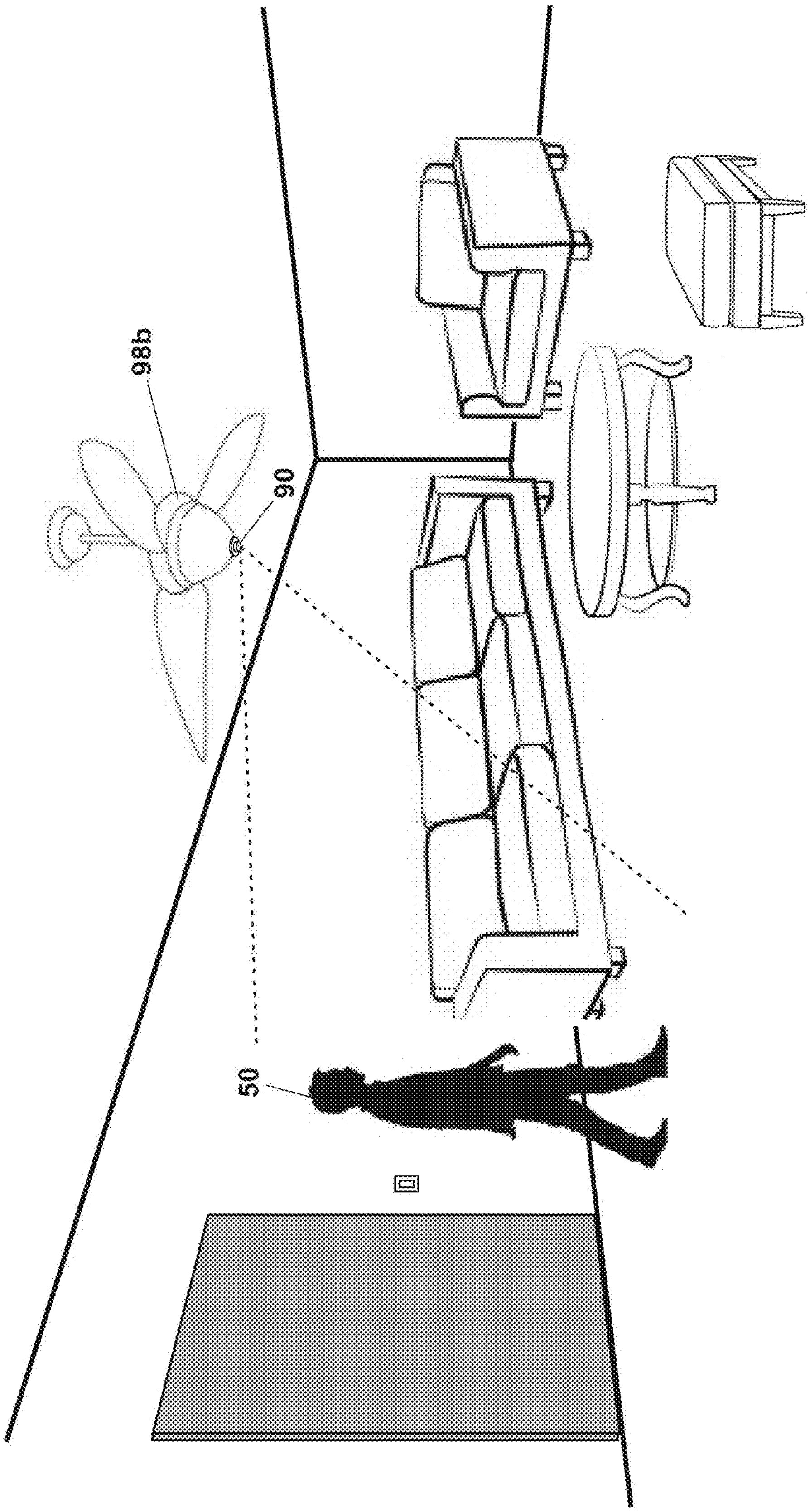


FIG. 39

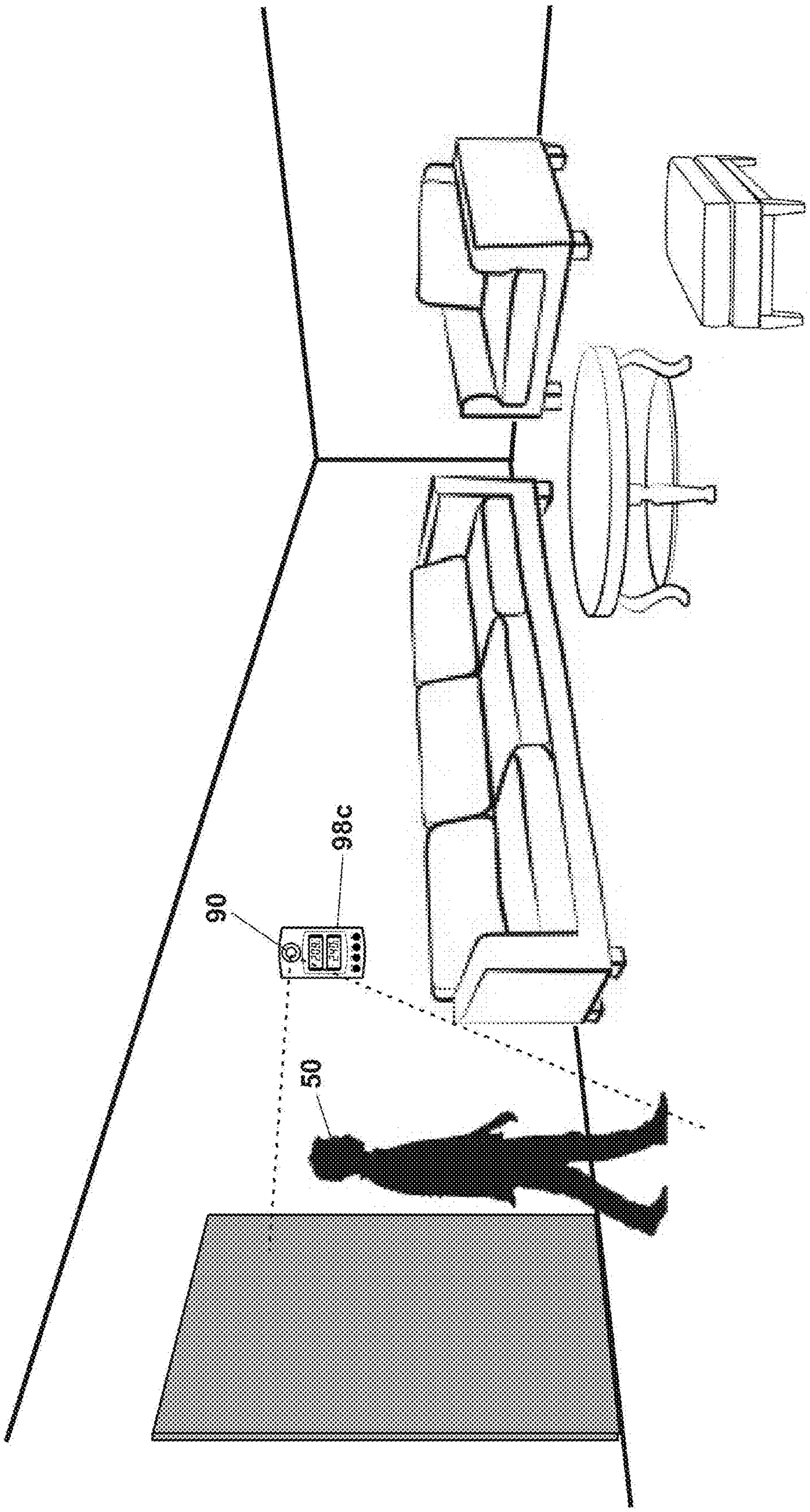


FIG. 40

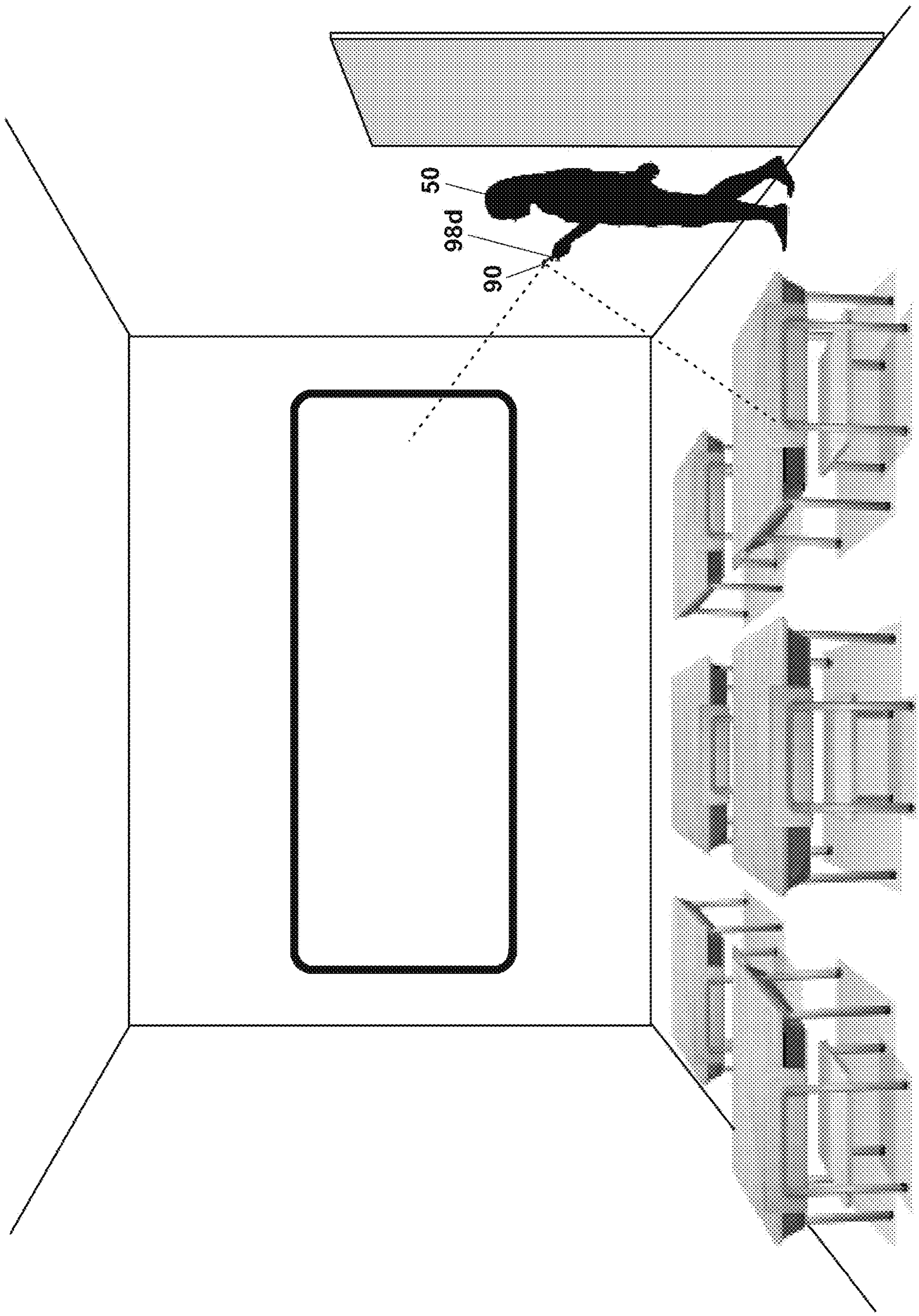


FIG. 41

MACHINE LEARNING FOR COMPUTING ENABLED SYSTEMS AND/OR DEVICES

CROSS REFERENCE TO RELATED APPLICATION

This application is a continuation of, and claims priority under 35 U.S.C. § 120 from, nonprovisional U.S. patent application Ser. No. 15/822,150 entitled "MACHINE LEARNING FOR COMPUTING ENABLED SYSTEMS AND/OR DEVICES", filed on Nov. 26, 2017. The disclosure of the foregoing document is incorporated herein by reference.

FIELD

The disclosure generally relates to computing enabled systems and/or devices.

COPYRIGHT NOTICE

BACKGROUND

Computing enabled systems and/or devices range from appliances, toys, entertainment electronics, computers, and communication systems and/or devices to vehicles, robots, and industrial systems and/or devices, and/or others. These systems and/or devices depend on user's input to various degrees for their operation. A machine learning solution is needed for computing enabled systems and/or devices to be less dependent on or fully independent from user input.

SUMMARY OF THE INVENTION

In some aspects, the disclosure relates to a system. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises a processor circuit configured to execute instruction sets for operating a device. The system may further include a memory unit configured to store data. The system may further include a picture capturing apparatus configured to capture digital pictures. The system may further include an artificial intelligence unit. In some embodiments, the artificial intelligence unit may be configured to: receive a first digital picture from the picture capturing apparatus. The artificial intelligence unit may be further configured to: receive one or more instruction sets for operating the device from the processor circuit. The artificial intelligence unit may be further configured to: learn the first digital picture correlated with the one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: receive a new digital picture from the picture capturing apparatus. The artificial intelligence unit may be further configured to: anticipate the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the one or more instruction sets

for operating the device correlated with the first digital picture, the executing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the one or more instruction sets for operating the device correlated with the first digital picture, the one or more operations performed in response to the executing by the processor circuit.

In certain embodiments, at least one of the processor circuit, the memory unit, the picture capturing apparatus, or the artificial intelligence unit are part of, operating on, or coupled to the device. In further embodiments, the device includes one or more devices. In further embodiments, the device includes a smartphone, a fixture, a control device, a computing enabled device, or a computer.

In some embodiments, the processor circuit includes one or more processor circuits. In further embodiments, the processor circuit includes a logic circuit. The logic circuit may include a microcontroller. The one or more instruction sets may include one or more inputs into or one or more outputs from the logic circuit.

In certain embodiments, the processor circuit includes a logic circuit, the instruction sets for operating the device include inputs into the logic circuit, and executing instruction sets for operating the device includes performing logic operations on the inputs into the logic circuit and producing outputs for operating the device. The logic circuit may include a microcontroller. In further embodiments, the processor circuit includes a logic circuit, the instruction sets for operating the device include outputs from the logic circuit for operating the device, and executing instruction sets for operating the device includes performing logic operations on inputs into the logic circuit and producing the outputs from the logic circuit for operating the device.

In some embodiments, the memory unit includes one or more memory units. In further embodiments, the memory unit resides on a remote computing device, the remote computing device coupled to the processor circuit via a network. The remote computing device may include a server.

In some embodiments, the picture capturing apparatus includes one or more picture capturing apparatuses. In further embodiments, the picture capturing apparatus includes a motion picture camera or a still picture camera. In further embodiments, the picture capturing apparatus resides on a remote device, the remote device coupled to the processor circuit via a network.

In certain embodiments, the artificial intelligence unit is coupled to the picture capturing apparatus. In further embodiments, the artificial intelligence unit is coupled to the memory unit. In further embodiments, the artificial intelligence unit is part of, operating on, or coupled to the processor circuit. In further embodiments, the system further comprises: a second processor circuit, wherein the artificial intelligence unit is part of, operating on, or coupled to the second processor circuit. In further embodiments, the artificial intelligence unit is part of, operating on, or coupled to a remote computing device, the remote computing device coupled to the processor circuit via a network. In further embodiments, the artificial intelligence unit includes a circuit, a computing apparatus, or a computing system attachable to the processor circuit. In further embodiments, the artificial intelligence unit includes a circuit, a computing apparatus, or a computing system attachable to the device. In further embodiments, the artificial intelligence unit is attachable to an application for operating the device, the application running on the processor circuit. In further embodiments, the artificial intelligence unit includes a circuit, a

computing apparatus, or a computing system built into the processor circuit. In further embodiments, the artificial intelligence unit includes a circuit, a computing apparatus, or a computing system built into the device. In further embodiments, the artificial intelligence unit is built into an application for operating the device, the application running on the processor circuit. In further embodiments, the artificial intelligence unit is provided as a feature of the processor circuit. In further embodiments, the artificial intelligence unit is provided as a feature of an application running on the processor circuit. In further embodiments, the artificial intelligence unit is provided as a feature of the device. In further embodiments, the artificial intelligence unit is further configured to: take control from, share control with, or release control to the processor circuit. In further embodiments, the artificial intelligence unit is further configured to: take control from, share control with, or release control to an application or an object of the application, the application running on the processor circuit. In further embodiments, the artificial intelligence unit is further configured to: take control from, share control with, or release control to a user or a system.

In some embodiments, the first digital picture includes a stream of digital pictures. In further embodiments, the new digital picture includes a stream of digital pictures. In further embodiments, the first and the new digital pictures portray the device's surrounding. In further embodiments, the first and the new digital pictures portray a remote device's surrounding. In further embodiments, the first or the new digital picture includes a JPEG picture, a GIF picture, a TIFF picture, a PNG picture, a PDF picture, or a digitally encoded picture. The stream of digital pictures may include a MPEG motion picture, an AVI motion picture, a FLV motion picture, a MOV motion picture, a RM motion picture, a SWF motion picture, a WMV motion picture, a DivX motion picture, or a digitally encoded motion picture. In further embodiments, the first digital picture includes a comparative digital picture whose at least one portion can be used for comparisons with at least one portion of digital pictures subsequent to the first digital picture, the digital pictures subsequent to the first digital picture comprising the new digital picture. In further embodiments, the first digital picture includes a comparative digital picture that can be used for comparisons with the new digital picture. In further embodiments, the new digital picture includes an anticipatory digital picture whose correlated one or more instruction sets can be used for anticipation of one or more instruction sets to be executed by the processor circuit.

In certain embodiments, the one or more instruction sets for operating the device include one or more instruction sets that temporally correspond to the first digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed at a time of the capturing the first digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed prior to the capturing the first digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed within a threshold period of time prior to the capturing the first digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed subsequent to the capturing the first digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed within a threshold period of time subsequent to the capturing the first digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed within a threshold period of time prior to the capturing the first digital picture or a threshold period of time subsequent to the capturing the first digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed from a start of capturing a preceding digital picture to a start of capturing the first digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed from a start of capturing the first digital picture to a start of capturing a subsequent digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed from a completion of capturing a preceding digital picture to a completion of capturing the first digital picture. The one or more instruction sets that temporally correspond to the first digital picture include one or more instruction sets executed from a completion of capturing the first digital picture to a completion of capturing a subsequent digital picture.

In some embodiments, the one or more instruction sets for operating the device are executed by the processor circuit. In further embodiments, the one or more instruction sets for operating the device are part of an application for operating the device, the application running on the processor circuit. In further embodiments, the one or more instruction sets for operating the device include one or more inputs into or one or more outputs from the processor circuit. In further embodiments, the one or more instruction sets for operating the device include values or states of one or more registers or elements of the processor circuit. In further embodiments, an instruction set includes at least one of: a command, a keyword, a symbol, an instruction, an operator, a variable, a value, an object, a data structure, a function, a parameter, a state, a signal, an input, an output, a character, a digit, or a reference thereto. In further embodiments, the one or more instruction sets include a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, a structured query language (SQL) code, or a machine code. In further embodiments, the one or more instruction sets include one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the processor circuit includes a logic circuit. The one or more instruction sets for operating the device include one or more inputs into a logic circuit. The one or more instruction sets for operating the device include one or more outputs from a logic circuit. In further embodiments, the one or more instruction sets for operating the device include one or more instruction sets for operating an application or an object of the application, the application running on the processor circuit.

In some embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes obtaining the one or more instruction sets from the processor circuit. In further embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes receiving the one or more instruction sets as they are executed by the processor circuit. In further embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes receiving the one or more instruction sets for operating the device from a register or an element of the processor circuit. In further embodiments, the

receiving the one or more instruction sets for operating the device from the processor circuit includes receiving the one or more instruction sets for operating the device from an element that is part of, operating on, or coupled to the processor circuit. In further embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes receiving the one or more instruction sets for operating the device from at least one of: the memory unit, the device, a virtual machine, a runtime engine, a hard drive, a storage device, a peripheral device, a network connected device, or a user. In further embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes receiving the one or more instruction sets from a plurality of processor circuits, applications, memory units, devices, virtual machines, runtime engines, hard drives, storage devices, peripheral devices, network connected devices, or users.

In certain embodiments, the processor circuit includes a logic circuit, and wherein the receiving the one or more instruction sets for operating the device from the processor circuit includes receiving the one or more instruction sets for operating the device from the logic circuit. The logic circuit may include a microcontroller. The receiving the one or more instruction sets for operating the device from the logic circuit may include receiving the one or more instruction sets for operating the device from an element of the logic circuit. The receiving the one or more instruction sets for operating the device from the logic circuit may include receiving one or more inputs into the logic circuit. The receiving the one or more instruction sets for operating the device from the logic circuit may include receiving one or more outputs from the logic circuit.

In some embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes receiving the one or more instruction sets for operating the device from an application for operating the device, the application running on the processor circuit. In further embodiments, the system further comprises: an application including instruction sets for operating the device, the application running on the processor circuit, wherein the receiving the one or more instruction sets for operating the device from the processor circuit includes receiving the one or more instruction sets for operating the device from the application.

In certain embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes receiving the one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes at least one of: tracing, profiling, or instrumentation of a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, a structured query language (SQL) code, or a machine code. In further embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes at least one of: tracing, profiling, or instrumentation of an element that is part of, operating on, or coupled to the processor circuit. In further embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes at least one of: tracing, profiling, or instrumentation of a register of the processor circuit, the memory unit, a storage, or a repository where the one or more instruction sets for operating the device are stored. In further embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes at least one of: tracing, profiling, or instrumentation of the processor circuit, the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a processing element. In further embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes at least one of: tracing, profiling, or instrumentation of the processor circuit or tracing, profiling, or instrumentation of a component of the processor circuit. In further embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes at least one of: tracing, profiling, or instrumentation of an application or an object of the application, the application running on the processor circuit. In further embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes at least one of: tracing, profiling, or instrumentation at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes at least one of: tracing, profiling, or instrumentation of one or more of code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes at least one of: tracing, profiling, or instrumentation of a user input. In further embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes at least one of: a manual, an automatic, a dynamic, or a just in time (JIT) tracing, profiling, or instrumentation. In further embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes utilizing at least one of: a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, a logging tool, or an independent tool for obtaining instruction sets. In further embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes utilizing an assembly language. In further embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes utilizing a branch or a jump. In further embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes a branch tracing or a simulation tracing.

In some embodiments, the system further comprises: an interface configured to receive instruction sets, wherein the one or more instruction sets for operating the device are received by the interface. The interface may include an acquisition interface.

In certain embodiments, the first digital picture correlated with the one or more instruction sets for operating the device includes a unit of knowledge of how the device operated in a visual surrounding. In further embodiments, the first digital picture correlated with the one or more instruction sets for operating the device is included in a neuron, a node, a vertex, or an element of a data structure. The data structure may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, or a knowledge structure. Some of the neurons, nodes, vertices, or elements may be interconnected.

In some embodiments, the first digital picture correlated with the one or more instruction sets for operating the device is structured into a knowledge cell. In further embodiments, the knowledge cell includes a unit of knowledge of how the

device operated in a visual surrounding. In further embodiments, the knowledge cell is included in a neuron, a node, a vertex, or an element of a data structure. The data structure may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, or a knowledge structure. Some of the neurons, nodes, vertices, or elements may be interconnected.

In certain embodiments, the learning the first digital picture correlated with the one or more instruction sets for operating the device includes correlating the first digital picture with the one or more instruction sets for operating the device. The correlating the first digital picture with the one or more instruction sets for operating the device may include generating a knowledge cell, the knowledge cell comprising the first digital picture correlated with the one or more instruction sets for operating the device. The correlating the first digital picture with the one or more instruction sets for operating the device may include structuring a unit of knowledge of how the device operated in a visual surrounding.

In some embodiments, the learning the first digital picture correlated with the one or more instruction sets for operating the device includes learning a user's knowledge, style, or methodology of operating the device in a visual surrounding. In further embodiments, the learning the first digital picture correlated with the one or more instruction sets for operating the device includes spontaneous learning the first digital picture correlated with the one or more instruction sets for operating the device.

In some embodiments, the learning the first digital picture correlated with the one or more instruction sets for operating the device includes storing, into the memory unit, the first digital picture correlated with the one or more instruction sets for operating the device, the first digital picture correlated with the one or more instruction sets for operating the device being part of a stored plurality of digital pictures correlated with one or more instruction sets for operating the device. In further embodiments, the plurality of digital pictures correlated with one or more instruction sets for operating the device include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, the plurality of digital pictures correlated with one or more instruction sets for operating the device are organized into a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, each of the plurality of digital pictures correlated with one or more instruction sets for operating the device is included in a neuron, a node, a vertex, or an element of a data structure. The data structure may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, or a knowledge structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the plurality of digital pictures correlated with one or more instruction sets for operating the device include a user's knowledge, style, or methodology of operating the device in visual surroundings. In further embodiments, the plurality of digital pictures correlated with one or more instruction sets for operating the device are stored on a remote computing device. In further embodiments, the plurality of digital pictures correlated with one or more instruction sets for operating the device include an artificial intelligence system for knowledge structuring, storing, or representation. The artificial intelligence system for knowledge structuring, storing, or representation may include at least one of: a deep learning system, a supervised learning system, an unsupervised learning system, a neural network, a search-based system, an optimization-based system, a logic-based system, a fuzzy logic-based system, a tree-based system, a graph-based system, a hierarchical system, a symbolic system, a sub-symbolic system, an evolutionary system, a genetic system, a multi-agent system, a deterministic system, a probabilistic system, or a statistical system.

In certain embodiments, the anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture includes comparing at least one portion of the new digital picture with at least one portion of the first digital picture. The at least one portion of the new digital picture may include at least one region, at least one feature, or at least one pixel of the new digital picture. The at least one portion of the first digital picture may include at least one region, at least one feature, or at least one pixel of the first digital picture. The comparing the at least one portion of the new digital picture with the at least one portion of the first digital picture may include comparing at least one region of the new digital picture with at least one region of the first digital picture. The comparing the at least one portion of the new digital picture with the at least one portion of the first digital picture may include comparing at least one feature of the new digital picture with at least one feature of the first digital picture. The at least one portion of the new digital picture with the at least one portion of the first digital picture may include comparing at least one pixel of the new digital picture with at least one pixel of the first digital picture. The comparing the at least one portion of the new digital picture with the at least one portion of the first digital picture may include at least one of: performing a color adjustment, performing a size adjustment, performing a content manipulation, utilizing a transparency, or utilizing a mask on the new or the first digital picture. The comparing the at least one portion of the new digital picture with the at least one portion of the first digital picture may include recognizing at least one person or object in the new digital picture and at least one person or object in the first digital picture, and comparing the at least one person or object from the new digital picture with the at least one person or object from the first digital picture.

In some embodiments, he anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture includes determining that there is at least a partial match between the new digital picture and the first digital picture. In further embodiments, the determining that there is at least a partial match between the new digital picture and the first digital picture includes determining that there is at least a partial match between one or more portions of the new digital picture and one or more portions of the first digital picture. In further embodiments, the determining that there is at least a partial match between the new digital picture and the first digital picture includes determining that a similarity between at least one portion of the new digital picture and at least one portion of the first digital picture exceeds a similarity threshold. In further embodiments, the determining that there is at least a partial match between the new digital picture and the first digital picture includes determining a substantial similarity between at least one portion of the new digital picture and at least one portion of the first digital picture. The at least one portion of the new digital picture may include at least one region, at least one feature, or at

least one pixel of the new digital picture. The at least one portion of the first digital picture may include at least one region, at least one feature, or at least one pixel of the first digital picture. The substantial similarity may be achieved when a similarity between the at least one portion of the new digital picture and the at least one portion of the first digital picture exceeds a similarity threshold. The substantial similarity may be achieved when a number or a percentage of matching or partially matching regions from the new digital picture and from the first digital picture exceeds a threshold number or threshold percentage. The substantial similarity may be achieved when a number or a percentage of matching or partially matching features from the new digital picture and from the first digital picture exceeds a threshold number or threshold percentage. The substantial similarity may be achieved when a number or a percentage of matching or partially matching pixels from the new digital picture and from the first digital picture exceeds a threshold number or threshold percentage. The substantial similarity may be achieved when one or more same or similar objects are recognized in the new digital picture and the first digital picture. In further embodiments, the determining that there is at least a partial match between the new digital picture and the first digital picture includes determining that a number or a percentage of matching regions from the new digital picture and from the first digital picture exceeds a threshold number or threshold percentage. The matching regions from the new digital picture and from the first digital picture may be determined factoring in at least one of: a location of a region, an importance of a region, a threshold for a similarity in a region, or a threshold for a difference in a region. In further embodiments, the determining that there is at least a partial match between the new digital picture and the first digital picture includes determining that a number or a percentage of matching features from the new digital picture and from the first digital picture exceeds a threshold number or threshold percentage. The matching features from the new digital picture and from the first digital picture may be determined factoring in at least one of: a type of a feature, an importance of a feature, a location of a feature, a threshold for a similarity in a feature, or a threshold for a difference in a feature. In further embodiments, the determining that there is at least a partial match between the new digital picture and the first digital picture includes determining that a number or a percentage of matching pixels from the new digital picture and from the first digital picture exceeds a threshold number or threshold percentage. The matching pixels from the new digital picture and from the first digital picture may be determined factoring in at least one of: a location of a pixel, a threshold for a similarity in a pixel, or a threshold for a difference in a pixel. In further embodiments, the determining that there is at least a partial match between the new digital picture and the first digital picture includes recognizing a same person or object in the new and the first digital pictures.

In some embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture instead of or prior to an instruction set that would have been executed next. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes modifying one or more instruction sets of the processor circuit. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes modifying a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes inserting the one or more instruction sets for operating the device correlated with the first digital picture into a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes redirecting the processor circuit to the one or more instruction sets for operating the device correlated with the first digital picture. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes redirecting the processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the one or more instruction sets for operating the device correlated with the first digital picture. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes transmitting, to the processor circuit for execution, the one or more instruction sets for operating the device correlated with the first digital picture. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes issuing an interrupt to the processor circuit and executing the one or more instruction sets for operating the device correlated with the first digital picture following the interrupt. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes modifying an element that is part of, operating on, or coupled to the processor circuit.

In certain embodiments, the processor circuit includes a logic circuit, and wherein the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes causing the logic circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture. The logic circuit may include a microcontroller. The causing the logic circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture may include modifying an element of the logic circuit. The causing the logic circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture may include inserting the one or more instruction sets for operating the device correlated with the first digital picture into an element of the logic circuit. The causing the logic circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture may include redirecting the logic circuit to the one or more instruction sets for operating the device correlated with the first digital picture. The causing the logic circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture may include replacing inputs into the logic circuit with the one or more instruction sets for operating the device correlated with the first digital picture. The causing the logic circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture may include replacing outputs from the logic circuit with the one or more instruction sets for operating the device correlated with the first digital picture.

In certain embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes causing an application for operating the device to execute the one or more instruction sets for operating the device correlated with the first digital picture, the application running on the processor circuit.

In further embodiments, the system further comprises: an application including instruction sets for operating the device, the application running on the processor circuit, wherein the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes modifying the application.

In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes redirecting an application to the one or more instruction sets for operating the device correlated with the first digital picture, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes redirecting an application to one or more alternate instruction sets, the application running on the processor circuit, the alternate instruction sets comprising the one or more instruction sets for operating the device correlated with the first digital picture. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes modifying one or more instruction sets of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes modifying at least one of: the memory unit, a register of the processor circuit, a storage, or a repository where instruction sets are stored or used. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes modifying one or more instruction sets for operating an application or an object of the application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes modifying at least one of: an element of the processor circuit, an element of the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes modifying one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes utilizing an assembly language. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes adding or inserting additional code into a code of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes at least one of: modifying, removing, rewriting, or overwriting a code of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes at least one of: branching, redirecting, extending, or hot swapping a code of an application, the application running on the processor circuit. The branching or redirecting the code may include inserting at least one of: a branch, a jump, or a means for redirecting an execution. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes implementing a user's knowledge, style, or methodology of operating the device in a visual surrounding.

In some embodiments, the system further comprises: an interface configured to cause execution of instruction sets, wherein the executing the one or more instruction sets for

operating the device correlated with the first digital picture is caused by the interface. The interface may include a modification interface.

In certain embodiments, the one or more operations defined by the one or more instruction sets for operating the device correlated with the first digital picture include at least one of: an operation with or by a smartphone, an operation with or by a fixture, an operation with or by a control device, or an operation with or by a computer or computing enabled device.

In some embodiments, the performing the one or more operations defined by the one or more instruction sets for operating the device correlated with the first digital picture includes implementing a user's knowledge, style, or methodology of operating the device in a visual surrounding.

In certain embodiments, the system further comprises: an application running on the processor circuit.

In some embodiments, the instruction sets for operating the device are part of an application for operating the device, the application running on the processor circuit.

In certain embodiments, the system further comprises: an application for operating the device, the application running on the processor circuit. The application for operating the device may include the instruction sets for operating the device.

In some embodiments, the artificial intelligence unit is further configured to: receive at least one extra information. In further embodiments, the at least one extra information include one or more of: a time information, a location information, a computed information, an observed information, a sensory information, or a contextual information. In further embodiments, the at least one extra information include one or more of: an information on a digital picture, an information on an object in the digital picture, an information on the device's visual surrounding, an information on an instruction set, an information on an application, an information on an object of the application, an information on the processor circuit, an information on the device, or an information on an user. In further embodiments, the artificial intelligence unit is further configured to: learn the first digital picture correlated with the at least one extra information. The learning the first digital picture correlated with at least one extra information may include correlating the first digital picture with the at least one extra information. The learning the first digital picture correlated with at least one extra information may include storing the first digital picture correlated with the at least one extra information into the memory unit. In further embodiments, the anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture includes anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between an extra information correlated with the new digital picture and an extra information correlated with the first digital picture. The anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between an extra information correlated with the new digital picture and an extra information correlated with the first digital picture may include comparing an extra information correlated with the new digital picture and an extra information correlated with the first digital picture. The anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between an extra information correlated with the new digital picture and an extra information correlated with the first digital picture may include determining that a similarity between an extra information correlated with the new digital picture and an extra information correlated with the first digital picture exceeds a similarity threshold.

In some embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: present, via the user interface, a user with an option to execute the one or more instruction sets for operating the device correlated with the first digital picture.

In certain embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: receive, via the user interface, a user's selection to execute the one or more instruction sets for operating the device correlated with the first digital picture.

In some embodiments, the artificial intelligence unit is further configured to: rate the executed one or more instruction sets for operating the device correlated with the first digital picture. The rating the executed one or more instruction sets for operating the device correlated with the first digital picture may include displaying, on a display, the executed one or more instruction sets for operating the device correlated with the first digital picture along with one or more rating values as options to be selected by a user. The rating the executed one or more instruction sets for operating the device correlated with the first digital picture may include rating the executed one or more instruction sets for operating the device correlated with the first digital picture without a user input. The rating the executed one or more instruction sets for operating the device correlated with the first digital picture may include associating one or more rating values with the executed one or more instruction sets for operating the device correlated with the first digital picture and storing the one or more rating values into the memory unit.

In certain embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: present, via the user interface, a user with an option to cancel the execution of the executed one or more instruction sets for operating the device correlated with the first digital picture. In further embodiments, the canceling the execution of the executed one or more instruction sets for operating the device correlated with the first digital picture includes restoring the processor circuit or the device to a prior state. The restoring the processor circuit or the device to a prior state may include saving the state of the processor circuit or the device prior to executing the one or more instruction sets for operating the device correlated with the first digital picture.

In some embodiments, the system further comprises: an input device configured to receive a user's operating directions, the user's operating directions for instructing the processor circuit on how to operate the device.

In certain embodiments, the autonomous device operating includes a partially or a fully autonomous device operating. The partially autonomous device operating may include executing the one or more instruction sets for operating the device correlated with the first digital picture responsive to a user confirmation. The fully autonomous device operating may include executing the one or more instruction sets for operating the device correlated with the first digital picture without a user confirmation.

In some embodiments, the artificial intelligence unit is further configured to: receive a second digital picture from the picture capturing apparatus; receive additional one or more instruction sets for operating the device from the processor circuit; and learn the second digital picture cor-

related with the additional one or more instruction sets for operating the device. In further embodiments, the second digital picture includes a second stream of digital pictures. In further embodiments, the learning the first digital picture correlated with the one or more instruction sets for operating the device and the learning the second digital picture correlated with the additional one or more instruction sets for operating the device include creating a connection between the first digital picture correlated with the one or more instruction sets for operating the device and the second digital picture correlated with the additional one or more instruction sets for operating the device. The connection may include or is associated with at least one of: an occurrence count, a weight, a parameter, or a data. In further embodiments, the learning the first digital picture correlated with the one or more instruction sets for operating the device and the learning the second digital picture correlated with the additional one or more instruction sets for operating the device include updating a connection between the first digital picture correlated with the one or more instruction sets for operating the device and the second digital picture correlated with the additional one or more instruction sets for operating the device. The updating the connection between the first digital picture correlated with the one or more instruction sets for operating the device and the second digital picture correlated with the additional one or more instruction sets for operating the device may include updating at least one of: an occurrence count, a weight, a parameter, or a data included in or associated with the connection. In further embodiments, the learning the first digital picture correlated with the one or more instruction sets for operating the device includes storing the first digital picture correlated with the one or more instruction sets for operating the device into a first node of a data structure, and wherein the learning the second digital picture correlated with the additional one or more instruction sets for operating the device includes storing the second digital picture correlated with the additional one or more instruction sets for operating the device into a second node of the data structure. The data structure may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, or a knowledge structure. The learning the first digital picture correlated with the one or more instruction sets for operating the device and the learning the second digital picture correlated with the additional one or more instruction sets for operating the device may include creating a connection between the first node and the second node. The learning the first digital picture correlated with the one or more instruction sets for operating the device and the learning the second digital picture correlated with the additional one or more instruction sets for operating the device may include updating a connection between the first node and the second node. In further embodiments, the first digital picture correlated with the one or more instruction sets for operating the device is stored into a first node of a neural network and the second digital picture correlated with the additional one or more instruction sets for operating the device is stored into a second node of the neural network. The first node and the second node may be connected by a connection. The first node may be part of a first layer of the neural network and the second node may be part of a second layer of the neural network. In further embodiments, the first digital picture correlated with the one or more instruction sets for operating the device is stored into a first node of a graph and the second digital picture correlated with the additional one or more instruction sets for operating the device is stored into a second node of the graph. The first node and the second node may be connected by a connection. In further embodiments, the first digital picture correlated with the one or more instruction sets for operating the device is stored into a first node of a sequence and the second digital picture correlated with the additional one or more instruction sets for operating the device is stored into a second node of the sequence.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first digital picture from a picture capturing apparatus. The operations may further include receiving one or more instruction sets for operating a device. The operations may further include learning the first digital picture correlated with the one or more instruction sets for operating the device. The operations may further include receiving a new digital picture from the picture capturing apparatus. The operations may further include anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture. The operations may further include causing an execution of the one or more instruction sets for operating the device correlated with the first digital picture, the causing performed in response to the anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture, wherein the device performs one or more operations defined by the one or more instruction sets for operating the device correlated with the first digital picture, the one or more operations performed in response to the executing.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first digital picture from a picture capturing apparatus by one or more processor circuits. The method may further include (b) receiving one or more instruction sets for operating a device by the one or more processor circuits. The method may further include (c) learning the first digital picture correlated with the one or more instruction sets for operating the device, the learning of (c) performed by the one or more processor circuits. The method may further include (d) receiving a new digital picture from the picture capturing apparatus by the one or more processor circuits. The method may further include (e) anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture, the anticipating of (e) performed by the one or more processor circuits. The method may further include (f) executing the one or more instruction sets for operating the device correlated with the first digital picture, the executing of (f) performed in response to the anticipating of (e). The method may further include (g) performing, by the device, one or more operations defined by the one or more instruction sets for operating the device correlated with the first digital picture, the one or more operations performed in response to the executing of (f).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems as applicable as well as the following embodiments.

In certain embodiments, the device includes one or more devices. In further embodiments, the device includes a smartphone, a fixture, a control device, a computing enabled device, or a computer. In further embodiments, the picture capturing apparatus includes one or more picture capturing apparatuses. In further embodiments, the picture capturing apparatus includes a motion picture camera or a still picture camera. In further embodiments, the picture capturing apparatus resides on a remote device, the remote device coupled to the one or more processor circuits via a network.

In some embodiments, the one or more instruction sets for operating the device include one or more instruction sets that temporally correspond to the first digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed at a time of the capturing the first digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed prior to the capturing the first digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed within a threshold period of time prior to the capturing the first digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed subsequent to the capturing the first digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed within a threshold period of time subsequent to the capturing the first digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed within a threshold period of time prior to the capturing the first digital picture or a threshold period of time subsequent to the capturing the first digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed from a start of capturing a preceding digital picture to a start of capturing the first digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed from a start of capturing the first digital picture to a start of capturing a subsequent digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed from a completion of capturing a preceding digital picture to a completion of capturing the first digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed from a completion of capturing the first digital picture to a completion of capturing a subsequent digital picture.

In certain embodiments, the one or more instruction sets for operating the device are executed by a processor circuit. In further embodiments, the one or more instruction sets for operating the device are part of an application for operating the device. In further embodiments, the one or more instruction sets for operating the device include one or more inputs into or one or more outputs from a processor circuit. In further embodiments, the one or more instruction sets for operating the device include values or states of one or more registers or elements of a processor circuit. In further embodiments, an instruction set includes at least one of: a command, a keyword, a symbol, an instruction, an operator, a variable, a value, an object, a data structure, a function, a parameter, a state, a signal, an input, an output, a character, a digit, or a reference thereto. In further embodiments, the

one or more instruction sets include a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, a structured query language (SQL) code, or a machine code. In further embodiments, the one or more instruction sets include one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the one or more instruction sets for operating the device include one or more inputs into a logic circuit. In further embodiments, the one or more instruction sets for operating the device include one or more outputs from a logic circuit. In further embodiments, the one or more instruction sets for operating the device include one or more instruction sets for operating an application or an object of the application.

In some embodiments, the receiving the one or more instruction sets for operating the device includes obtaining the one or more instruction sets. In further embodiments, the receiving the one or more instruction sets for operating the device includes receiving the one or more instruction sets as they are executed. In further embodiments, the receiving the one or more instruction sets for operating the device includes receiving the one or more instruction sets for operating the device from a register or an element of a processor circuit. In further embodiments, the receiving the one or more instruction sets for operating the device includes receiving the one or more instruction sets for operating the device from an element that is part of, operating on, or coupled to a processor circuit. In further embodiments, the receiving the one or more instruction sets for operating the device includes receiving the one or more instruction sets for operating the device from at least one of: a memory unit, the device, a virtual machine, a runtime engine, a hard drive, a storage device, a peripheral device, a network connected device, or a user. In further embodiments, the receiving the one or more instruction sets for operating the device includes receiving the one or more instruction sets from a plurality of processor circuits, applications, memory units, devices, virtual machines, runtime engines, hard drives, storage devices, peripheral devices, network connected devices, or users.

In certain embodiments, the receiving the one or more instruction sets for operating the device includes receiving the one or more instruction sets for operating the device from a logic circuit. The logic circuit may include a microcontroller. The receiving the one or more instruction sets for operating the device from the logic circuit may include receiving the one or more instruction sets for operating the device from an element of the logic circuit. The receiving the one or more instruction sets for operating the device from the logic circuit may include receiving one or more inputs into the logic circuit. The receiving the one or more instruction sets for operating the device from the logic circuit may include receiving one or more outputs from the logic circuit.

In some embodiments, the receiving the one or more instruction sets for operating the device includes receiving the one or more instruction sets for operating the device from an application for operating the device. In further embodiments, the receiving the one or more instruction sets for operating the device includes receiving the one or more instruction sets for operating the device from an application, the application including instruction sets for operating the device. In further embodiments, the receiving the one or more instruction sets for operating the device includes receiving the one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In

further embodiments, the receiving the one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, a structured query language (SQL) code, or a machine code. In further embodiments, the receiving the one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of an element that is part of, operating on, or coupled to a processor circuit. In further embodiments, the receiving the one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a register of a processor circuit, a memory unit, a storage, or a repository where the one or more instruction sets for operating the device are stored. In further embodiments, the receiving the one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a processor circuit, the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a processing element. In further embodiments, the receiving the one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a processor circuit or tracing, profiling, or instrumentation of a component of the processor circuit. In further embodiments, the receiving the one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of an application or an object of the application. In further embodiments, the receiving the one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the receiving the one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of one or more of code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the receiving the one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a user input. In further embodiments, the receiving the one or more instruction sets for operating the device includes at least one of: a manual, an automatic, a dynamic, or a just in time (JIT) tracing, profiling, or instrumentation. In further embodiments, the receiving the one or more instruction sets for operating the device includes utilizing at least one of: a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, a logging tool, or an independent tool for obtaining instruction sets. In further embodiments, the receiving the one or more instruction sets for operating the device includes utilizing an assembly language. In further embodiments, the receiving the one or more instruction sets for operating the device includes utilizing a branch or a jump. In further embodiments, the receiving the one or more instruction sets for operating the device includes a branch tracing or a simulation tracing. In further embodiments, the receiving the one or more instruction sets for operating the device includes receiving the one or more instruction sets for operating the device by an interface. The interface may include an acquisition interface.

In certain embodiments, the first digital picture correlated with the one or more instruction sets for operating the device includes a unit of knowledge of how the device operated in a visual surrounding. In further embodiments, the first digital picture correlated with the one or more instruction sets for operating the device is included in a neuron, a node, a vertex, or an element of a data structure. The data structure may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, or a knowledge structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the first digital picture correlated with the one or more instruction sets for operating the device is structured into a knowledge cell. In further embodiments, the knowledge cell includes a unit of knowledge of how the device operated in a visual surrounding. In further embodiments, the knowledge cell is included in a neuron, a node, a vertex, or an element of a data structure. The data structure may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, or a knowledge structure. Some of the neurons, nodes, vertices, or elements may be interconnected.

In certain embodiments, the learning the first digital picture correlated with the one or more instruction sets for operating the device includes correlating the first digital picture with the one or more instruction sets for operating the device. The correlating the first digital picture with the one or more instruction sets for operating the device may include generating a knowledge cell, the knowledge cell comprising the first digital picture correlated with the one or more instruction sets for operating the device. The correlating the first digital picture with the one or more instruction sets for operating the device may include structuring a unit of knowledge of how the device operated in a visual surrounding. In further embodiments, the learning the first digital picture correlated with the one or more instruction sets for operating the device includes learning a user's knowledge, style, or methodology of operating the device in a visual surrounding. In further embodiments, the learning the first digital picture correlated with the one or more instruction sets for operating the device includes spontaneous learning the first digital picture correlated with the one or more instruction sets for operating the device.

In some embodiments, the learning the first digital picture correlated with the one or more instruction sets for operating the device includes storing, into a memory unit, the first digital picture correlated with the one or more instruction sets for operating the device, the first digital picture correlated with the one or more instruction sets for operating the device being part of a stored plurality of digital pictures correlated with one or more instruction sets for operating the device. In further embodiments, the memory unit includes one or more memory units. In further embodiments, the memory unit resides on a remote computing device, the remote computing device coupled to the one or more processor circuits via a network. The remote computing device may include a server. In further embodiments, the plurality of digital pictures correlated with one or more instruction sets for operating the device include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, the plurality of digital pictures correlated with one or more instruction sets for operating the device are organized into a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, each of the plurality of digital pictures correlated with one or more instruction sets for operating the device is included in a neuron, a node, a vertex, or an element of a data structure. The data structure may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowl-

edge cells, a knowledgebase, or a knowledge structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the plurality of digital pictures correlated with one or more instruction sets for operating the device include a user's knowledge, style, or methodology of operating the device in visual surroundings. In further embodiments, the plurality of digital pictures correlated with one or more instruction sets for operating the device are stored on a remote computing device. In further embodiments, the plurality of digital pictures correlated with one or more instruction sets for operating the device include an artificial intelligence system for knowledge structuring, storing, or representation. The artificial intelligence system for knowledge structuring, storing, or representation may include at least one of: a deep learning system, a supervised learning system, an unsupervised learning system, a neural network, a search-based system, an optimization-based system, a logic-based system, a fuzzy logic-based system, a tree-based system, a graph-based system, a hierarchical system, a symbolic system, a sub-symbolic system, an evolutionary system, a genetic system, a multi-agent system, a deterministic system, a probabilistic system, or a statistical system.

In some embodiments, the anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture includes comparing at least one portion of the new digital picture with at least one portion of the first digital picture. The at least one portion of the new digital picture may include at least one region, at least one feature, or at least one pixel of the new digital picture. The at least one portion of the first digital picture may include at least one region, at least one feature, or at least one pixel of the first digital picture. The comparing the at least one portion of the new digital picture with the at least one portion of the first digital picture may include comparing at least one region of the new digital picture with at least one region of the first digital picture. The comparing the at least one portion of the new digital picture with the at least one portion of the first digital picture may include comparing at least one feature of the new digital picture with at least one feature of the first digital picture. The comparing the at least one portion of the new digital picture with the at least one portion of the first digital picture may include comparing at least one pixel of the new digital picture with at least one pixel of the first digital picture. The comparing the at least one portion of the new digital picture with the at least one portion of the first digital picture may include at least one of: performing a color adjustment, performing a size adjustment, performing a content manipulation, utilizing a transparency, or utilizing a mask on the new or the first digital picture. The comparing the at least one portion of the new digital picture with the at least one portion of the first digital picture may include recognizing at least one person or object in the new digital picture and at least one person or object in the first digital picture, and comparing the at least one person or object from the new digital picture with the at least one person or object from the first digital picture.

In certain embodiments, the anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture includes determining that there is at least a partial match between the new digital picture and the first digital picture. In further embodiments, the determining that there is at least a partial match between the new digital picture and the first digital picture includes determining that there is at least a partial match between one or more portions of the new digital picture and one or more portions of the first digital picture. In further embodiments, the determining that there is at least a partial match between the new digital picture and the first digital picture includes determining that a similarity between at least one portion of the new digital picture and at least one portion of the first digital picture exceeds a similarity threshold. In further embodiments, the determining that there is at least a partial match between the new digital picture and the first digital picture includes determining a substantial similarity between at least one portion of the new digital picture and at least one portion of the first digital picture. The at least one portion of the new digital picture may include at least one region, at least one feature, or at least one pixel of the new digital picture. The at least one portion of the first digital picture may include at least one region, at least one feature, or at least one pixel of the first digital picture. The substantial similarity may be achieved when a similarity between the at least one portion of the new digital picture and the at least one portion of the first digital picture exceeds a similarity threshold. The substantial similarity may be achieved when a number or a percentage of matching or partially matching regions from the new digital picture and from the first digital picture exceeds a threshold number or threshold percentage. The substantial similarity may be achieved when a number or a percentage of matching or partially matching features from the new digital picture and from the first digital picture exceeds a threshold number or threshold percentage. The substantial similarity may be achieved when a number or a percentage of matching or partially matching pixels from the new digital picture and from the first digital picture exceeds a threshold number or threshold percentage. The substantial similarity may be achieved when one or more same or similar objects are recognized in the new digital picture and the first digital picture. In further embodiments, the determining that there is at least a partial match between the new digital picture and the first digital picture includes determining that a number or a percentage of matching regions from the new digital picture and from the first digital picture exceeds a threshold number or threshold percentage. The matching regions from the new digital picture and from the first digital picture may be determined factoring in at least one of: a location of a region, an importance of a region, a threshold for a similarity in a region, or a threshold for a difference in a region. In further embodiments, the determining that there is at least a partial match between the new digital picture and the first digital picture includes determining that a number or a percentage of matching features from the new digital picture and from the first digital picture exceeds a threshold number or threshold percentage. The matching features from the new digital picture and from the first digital picture may be determined factoring in at least one of: a type of a feature, an importance of a feature, a location of a feature, a threshold for a similarity in a feature, or a threshold for a difference in a feature. In further embodiments, the determining that there is at least a partial match between the new digital picture and the first digital picture includes determining that a number or a percentage of matching pixels from the new digital picture and from the first digital picture exceeds a threshold number or threshold percentage. The matching pixels from the new digital picture and from the first digital picture may be determined factoring in at least one of: a location of a pixel, a threshold for a similarity in a pixel, or a threshold for a difference in a pixel. In further embodiments, the determining that there is at least a partial match

match between the new digital picture and the first digital picture includes recognizing a same person or object in the new and the first digital pictures.

In some embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes executing the one or more instruction sets for operating the device correlated with the first digital picture instead of or prior to an instruction set that would have been executed next. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes modifying one or more instruction sets. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes modifying a register or an element of a processor circuit. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes inserting the one or more instruction sets for operating the device correlated with the first digital picture into a register or an element of a processor circuit. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes redirecting a processor circuit to the one or more instruction sets for operating the device correlated with the first digital picture. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes redirecting a processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the one or more instruction sets for operating the device correlated with the first digital picture. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes transmitting, to a processor circuit for execution, the one or more instruction sets for operating the device correlated with the first digital picture. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes issuing an interrupt to a processor circuit and executing the one or more instruction sets for operating the device correlated with the first digital picture following the interrupt. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes modifying an element that is part of, operating on, or coupled to a processor circuit.

In certain embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes executing, by a logic circuit, the one or more instruction sets for operating the device correlated with the first digital picture. The logic circuit may include a microcontroller. The executing, by the logic circuit, the one or more instruction sets for operating the device correlated with the first digital picture may include modifying an element of the logic circuit. The executing, by the logic circuit, the one or more instruction sets for operating the device correlated with the first digital picture may include inserting the one or more instruction sets for operating the device correlated with the first digital picture into an element of the logic circuit. The executing, by the logic circuit, the one or more instruction sets for operating the device correlated with the first digital picture may include redirecting the logic circuit to the one or more instruction sets for operating the device correlated with the first digital picture. The executing, by the logic circuit, the one or more instruction sets for operating the device correlated with the first digital picture may include replacing inputs into the logic circuit with the one or more instruction sets for operating the device correlated with the first digital picture. The executing, by the logic circuit, the one or more instruction sets for operating the device correlated with the first digital picture may include replacing outputs from the logic circuit with the one or more instruction sets for operating the device correlated with the first digital picture.

In some embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes executing, by an application for operating the device, the one or more instruction sets for operating the device correlated with the first digital picture. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes modifying an application, the application including instruction sets for operating the device. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes redirecting an application to the one or more instruction sets for operating the device correlated with the first digital picture. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes redirecting an application to one or more alternate instruction sets, the alternate instruction sets comprising the one or more instruction sets for operating the device correlated with the first digital picture. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes modifying one or more instruction sets of an application. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes modifying at least one of: a memory unit, a register of a processor circuit, a storage, or a repository where instruction sets are stored or used. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes modifying one or more instruction sets for operating an application or an object of the application. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes modifying at least one of: an element of a processor circuit, an element of the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes modifying one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of an application. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes utilizing one or more of

a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes utilizing an assembly language. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes adding or inserting additional code into a code of an application. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes at least one of: modifying, removing, rewriting, or overwriting a code of an application. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes at least one of: branching, redirecting, extending, or hot swapping a code of an application. The branching or redirecting the code may include inserting at least one of: a branch, a jump, or a means for redirecting an execution. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes implementing a user's knowledge, style, or methodology of operating the device in a visual surrounding. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes executing the one or more instruction sets for operating the device correlated with the first digital picture via an interface. The interface may include a modification interface.

In certain embodiments, the one or more operations defined by the one or more instruction sets for operating the device correlated with the first digital picture include at least one of: an operation with or by a smartphone, an operation with or by a fixture, an operation with or by a control device, or an operation with or by a computer or computing enabled device. In further embodiments, the performing the one or more operations defined by the one or more instruction sets for operating the device correlated with the first digital picture includes implementing a user's knowledge, style, or methodology of operating the device in a visual surrounding.

In some embodiments, the instruction sets for operating the device are part of an application for operating the device.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving at least one extra information. In further embodiments, the at least one extra information include one or more of: a time information, a location information, a computed information, an observed information, a sensory information, or a contextual information. In further embodiments, the at least one extra information include one or more of: an information on a digital picture, an information on an object in the digital picture, an information on the device's visual surrounding, an information on an instruction set, an information on an application, an information on an object of the application, an information on a processor circuit, an information on the device, or an information on an user. In further embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: learning the first digital picture correlated with the at least one extra information. The learning the first digital picture correlated with at least one extra information may include correlating the first digital picture with the at least one extra information. The learning the first digital picture correlated with at least one extra information may include storing the first digital picture correlated with the at least one extra information into a memory unit. In further embodiments, the anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture includes anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between an extra information correlated with the new digital picture and an extra information correlated with the first digital picture. The anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between an extra information correlated with the new digital picture and an extra information correlated with the first digital picture may include comparing an extra information correlated with the new digital picture and an extra information correlated with the first digital picture. The anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between an extra information correlated with the new digital picture and an extra information correlated with the first digital picture may include determining that a similarity between an extra information correlated with the new digital picture and an extra information correlated with the first digital picture exceeds a similarity threshold.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: presenting, via a user interface, a user with an option to execute the one or more instruction sets for operating the device correlated with the first digital picture.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving, via a user interface, a user's selection to execute the one or more instruction sets for operating the device correlated with the first digital picture.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: rating the executed one or more instruction sets for operating the device correlated with the first digital picture. The rating the executed one or more instruction sets for operating the device correlated with the first digital picture may include displaying, on a display, the executed one or more instruction sets for operating the device correlated with the first digital picture along with one or more

rating values as options to be selected by a user. The rating the executed one or more instruction sets for operating the device correlated with the first digital picture may include rating the executed one or more instruction sets for operating the device correlated with the first digital picture without a user input. The rating the executed one or more instruction sets for operating the device correlated with the first digital picture may include associating one or more rating values with the executed one or more instruction sets for operating the device correlated with the first digital picture and storing the one or more rating values into a memory unit.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: presenting, via a user interface, a user with an option to cancel the execution of the executed one or more instruction sets for operating the device correlated with the first digital picture. In further embodiments, the canceling the execution of the executed one or more instruction sets for operating the device correlated with the first digital picture includes restoring a processor circuit or the device to a prior state. The restoring the processor circuit or the device to a prior state may include saving the state of the processor circuit or the device prior to executing the one or more instruction sets for operating the device correlated with the first digital picture.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving, via an input device, a user's operating directions, the user's operating directions for instructing a processor circuit on how to operate the device.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving a second digital picture from the picture capturing apparatus; receiving additional one or more instruction sets for operating the device; and learning the second digital picture correlated with the additional one or more instruction sets for operating the device. In further embodiments, the second digital picture includes a second stream of digital pictures. In further embodiments, the learning the first digital picture correlated with the one or more instruction sets for operating the device and the learning the second digital picture correlated with the additional one or more instruction sets for operating the device include creating a connection between the first digital picture correlated with the one or more instruction sets for operating the device and the second digital picture correlated with the additional one or more instruction sets for operating the device. The connection may include or is associated with at least one of: an occurrence count, a weight, a parameter, or a data. In further embodiments, the learning the first digital picture correlated with the one or more instruction sets for operating the device and the learning the second digital picture correlated with the additional one or more instruction sets for operating the device include updating a connection between the first digital picture correlated with the one or more instruction sets for operating the device and the second digital picture correlated with the additional one or more instruction sets for operating the device. The updating the connection between the first digital picture correlated with the one or more instruction sets for operating the device and the second digital picture correlated with the additional one or more instruction sets for operating the device may include updating at least one of: an occurrence count, a weight, a parameter, or a data included in or associated with the connection. In further embodiments, the learning the first digital picture correlated with the one or more instruction sets for operating the device includes storing the first digital picture correlated with the one or more instruction sets for operating the device into a first node of a data structure, and wherein the learning the second digital picture correlated with the additional one or more instruction sets for operating the device may include storing the second digital picture correlated with the additional one or more instruction sets for operating the device into a second node of the data structure. The data structure may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, or a knowledge structure. The learning the first digital picture correlated with the one or more instruction sets for operating the device and the learning the second digital picture correlated with the additional one or more instruction sets for operating the device may include creating a connection between the first node and the second node. The learning the first digital picture correlated with the one or more instruction sets for operating the device and the learning the second digital picture correlated with the additional one or more instruction sets for operating the device may include updating a connection between the first node and the second node. In further embodiments, the first digital picture correlated with the one or more instruction sets for operating the device is stored into a first node of a neural network and the second digital picture correlated with the additional one or more instruction sets for operating the device is stored into a second node of the neural network. The first node and the second node may be connected by a connection. The first node may be part of a first layer of the neural network and the second node may be part of a second layer of the neural network. In further embodiments, the first digital picture correlated with the one or more instruction sets for operating the device is stored into a first node of a graph and the second digital picture correlated with the additional one or more instruction sets for operating the device is stored into a second node of the graph. The first node and the second node may be connected by a connection. In further embodiments, the first digital picture correlated with the one or more instruction sets for operating the device is stored into a first node of a sequence and the second digital picture correlated with the additional one or more instruction sets for operating the device is stored into a second node of the sequence.

In some aspects, the disclosure relates to a system for learning a visual surrounding for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises a processor circuit configured to execute instruction sets for operating a device. The system may further include a memory unit configured to store data. The system may further include a picture capturing apparatus configured to capture digital pictures. The system may further include an artificial intelligence unit. In some embodiments, the artificial intelligence unit may be configured to: receive a first digital picture from the picture capturing apparatus. The artificial intelligence unit may be further configured to: receive one or more instruction sets for operating the device from the processor circuit. The artificial intelligence unit may be further configured to: learn the first digital picture correlated with the one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first digital picture from a picture capturing

apparatus. The operations may further include: receiving one or more instruction sets for operating a device. The operations may further include: learning the first digital picture correlated with the one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first digital picture from a picture capturing apparatus by one or more processor circuits. The method may further include: (b) receiving one or more instruction sets for operating a device by the one or more processor circuits. The method may further include: (c) learning the first digital picture correlated with the one or more instruction sets for operating the device, the learning of (c) performed by the one or more processor circuits.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems as applicable.

In some aspects, the disclosure relates to a system for using a visual surrounding for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises a processor circuit configured to execute instruction sets for operating a device. The system may further include a memory unit configured to store data. The system may further include a picture capturing apparatus configured to capture digital pictures. The system may further include an artificial intelligence unit. In some embodiments, the artificial intelligence unit may be configured to: access the memory unit that stores a plurality of digital pictures correlated with one or more instruction sets for operating the device, the plurality including a first digital picture correlated with one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: receive a new digital picture from the picture capturing apparatus. The artificial intelligence unit may be further configured to: anticipate the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture, the executing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the one or more instruction sets for operating the device correlated with the first digital picture, the one or more operations performed in response to the executing by the processor circuit.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: accessing a memory unit that stores a plurality of digital pictures correlated with one or more instruction sets for operating a device, the plurality including a first digital picture correlated with one or more instruction sets for operating the device. The operations may further include: receiving a new digital picture from a picture capturing apparatus. The operations may further include: anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture. The operations may further include: causing an execution of the one or more instruction sets for operating the device correlated with the first digital picture, the causing performed in response to the anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture, wherein the device performs one or more operations defined by the one or more instruction sets for operating the device correlated with the first digital picture, the one or more operations performed in response to the executing.

In some aspects, the disclosure relates to a method comprising: (a) accessing a memory unit that stores a plurality of digital pictures correlated with one or more instruction sets for operating a device, the plurality including a first digital picture correlated with one or more instruction sets for operating the device, the accessing of (a) performed by the one or more processor circuits. The method may further include: (b) receiving a new digital picture from a picture capturing apparatus by the one or more processor circuits. The method may further include: (c) anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture, the anticipating of (c) performed by the one or more processor circuits. The method may further include: (d) executing the one or more instruction sets for operating the device correlated with the first digital picture, the executing of (d) performed in response to the anticipating of (c). The method may further include: (e) performing, by the device, one or more operations defined by the one or more instruction sets for operating the device correlated with the first digital picture, the one or more operations performed in response to the executing of (d).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems as applicable.

In some aspects, the disclosure relates to a system for learning and using a visual surrounding for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises a processor circuit configured to execute instruction sets for operating a device. The system may further include a memory unit configured to store data. The system may further include a picture capturing apparatus configured to capture digital pictures. The system may further include an artificial intelligence. In some embodiments, the artificial intelligence unit may be configured to: receive a first stream of digital pictures from the picture capturing apparatus. The artificial intelligence unit may be further configured to: receive one or more instruction sets for operating the device from the processor circuit. The artificial intelligence unit may be further configured to: learn the first stream of digital pictures correlated with the one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: receive a new stream of digital pictures from the picture capturing apparatus. The artificial intelligence unit may be further configured to: anticipate the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between the new stream of digital pictures and the first stream of digital pictures. The artificial intelligence unit may be further

configured to: cause the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures, the executing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the one or more instruction sets for operating the device correlated with the first stream of digital pictures, the one or more operations performed in response to the executing by the processor circuit.

In certain embodiments, the first stream of digital pictures includes one or more digital pictures. In further embodiments, the new stream of digital pictures includes one or more digital pictures. In further embodiments, the first and the new streams of digital pictures portray the device's surrounding. In further embodiments, the first and the new streams of digital pictures portray a remote device's surrounding. In further embodiments, the first or the new stream of digital pictures includes a digital motion picture. The digital motion picture may include a MPEG motion picture, an AVI motion picture, a FLV motion picture, a MOV motion picture, a RM motion picture, a SWF motion picture, a WMV motion picture, a DivX motion picture, or a digitally encoded motion picture. In further embodiments, the first stream of digital pictures includes a comparative stream of digital pictures whose at least one portion can be used for comparisons with at least one portion of streams of digital pictures subsequent to the first stream of digital pictures, the streams of digital pictures subsequent to the first stream of digital pictures comprising the new stream of digital pictures. In further embodiments, the first stream of digital pictures includes a comparative stream of digital pictures that can be used for comparisons with the new stream of digital pictures. In further embodiments, the new stream of digital pictures includes an anticipatory stream of digital pictures whose correlated one or more instruction sets can be used for anticipation of one or more instruction sets to be executed by the processor circuit.

In some embodiments, the first stream of digital pictures correlated with the one or more instruction sets for operating the device includes a unit of knowledge of how the device operated in a visual surrounding. In further embodiments, the first stream of digital pictures correlated with the one or more instruction sets for operating the device is included in a neuron, a node, a vertex, or an element of a data structure. In further embodiments, the data structure includes a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, or a knowledge structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the first stream of digital pictures correlated with the one or more instruction sets for operating the device is structured into a knowledge cell. In further embodiments, the knowledge cell includes a unit of knowledge of how the device operated in a visual surrounding. In further embodiments, the knowledge cell is included in a neuron, a node, a vertex, or an element of a data structure. The data structure may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, or a knowledge structure. Some of the neurons, nodes, vertices, or elements may be interconnected.

In certain embodiments, the learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device includes correlating the first stream of digital pictures with the one or more instruction sets for operating the device. The correlating the first stream of digital pictures with the one or more instruction sets for operating the device may include generating a knowledge cell, the knowledge cell comprising the first stream of digital pictures correlated with the one or more instruction sets for operating the device. The correlating the first stream of digital pictures with the one or more instruction sets for operating the device may include structuring a unit of knowledge of how the device operated in a visual surrounding. In further embodiments, the learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device includes learning a user's knowledge, style, or methodology of operating the device in a visual surrounding. In further embodiments, the learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device includes spontaneous learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device.

In some embodiments, the learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device includes storing, into the memory unit, the first stream of digital pictures correlated with the one or more instruction sets for operating the device, the first stream of digital pictures correlated with the one or more instruction sets for operating the device being part of a stored plurality of streams of digital pictures correlated with one or more instruction sets for operating the device. In further embodiments, the plurality of streams of digital pictures correlated with one or more instruction sets for operating the device include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, the plurality of streams of digital pictures correlated with one or more instruction sets for operating the device are organized into a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, each of the plurality of streams of digital pictures correlated with one or more instruction sets for operating the device is included in a neuron, a node, a vertex, or an element of a data structure. The data structure may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, or a knowledge structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the plurality of streams of digital pictures correlated with one or more instruction sets for operating the device include a user's knowledge, style, or methodology of operating the device in visual surroundings. In further embodiments, the plurality of streams of digital pictures correlated with one or more instruction sets for operating the device are stored on a remote computing device. In further embodiments, the plurality of streams of digital pictures correlated with one or more instruction sets for operating the device include an artificial intelligence system for knowledge structuring, storing, or representation. The artificial intelligence system for knowledge structuring, storing, or representation may include at least one of: a deep learning system, a supervised learning system, an unsupervised learning system, a neural network, a search-based system, an optimization-based system, a logic-based system, a fuzzy logic-based system, a tree-based system, a graph-based system, a hierarchical system, a symbolic system, a sub-symbolic system, an evolutionary system, a genetic system, a multi-agent system, a deterministic system, a probabilistic system, or a statistical system.

In some embodiments, the anticipating the one or more instruction sets for operating the device correlated with the

first stream of digital pictures based on at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes comparing at least one portion of the new stream of digital pictures with at least one portion of the first stream of digital pictures. The at least one portion of the new stream of digital pictures may include at least one digital picture, at least one region, at least one feature, or at least one pixel of the new stream of digital pictures. The at least one portion of the first stream of digital pictures may include at least one digital picture, at least one region, at least one feature, or at least one pixel of the first stream of digital pictures. The comparing the at least one portion of the new stream of digital pictures with the at least one portion of the first stream of digital pictures may include comparing at least one digital picture of the new stream of digital pictures with at least one digital picture of the first stream of digital pictures. The at least one portion of the new stream of digital pictures with the at least one portion of the first stream of digital pictures may include comparing at least one region of at least one digital picture of the new stream of digital pictures with at least one region of at least one digital picture of the first stream of digital pictures. The comparing the at least one portion of the new stream of digital pictures with the at least one portion of the first stream of digital pictures may include comparing at least one feature of at least one digital picture of the new stream of digital pictures with at least one feature of at least one digital picture of the first stream of digital pictures. The comparing the at least one portion of the new stream of digital pictures with the at least one portion of the first stream of digital pictures may include comparing at least one pixel of at least one digital picture of the new stream of digital pictures with at least one pixel of at least one digital picture of the first stream of digital pictures. The comparing the at least one portion of the new stream of digital pictures with the at least one portion of the first stream of digital pictures may include at least one of: performing a color adjustment, performing a size adjustment, performing a content manipulation, performing temporal alignment, performing dynamic time warping, utilizing a transparency, or utilizing a mask on the new or the first stream of digital pictures. The comparing the at least one portion of the new stream of digital pictures with the at least one portion of the first stream of digital pictures may include recognizing at least one person or object in the new stream of digital pictures and at least one person or object in the first stream of digital pictures, and comparing the at least one person or object from the new stream of digital pictures with the at least one person or object from the first stream of digital pictures.

In certain embodiments, the anticipating the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes determining that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures. In further embodiments, the determining that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes determining that there is at least a partial match between one or more portions of the new stream of digital pictures and one or more portions of the first stream of digital pictures. In further embodiments, the determining that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes determining that a similarity between at least one portion of the new stream of digital pictures and at least one portion of the first stream of digital pictures exceeds a similarity threshold. In further embodiments, the determining that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes determining a substantial similarity between at least one portion of the new stream of digital pictures and at least one portion of the first stream of digital pictures. The at least one portion of the new stream of digital pictures may include at least one digital picture, at least one region, at least one feature, or at least one pixel of the new stream of digital pictures. The at least one portion of the first stream of digital pictures may include at least one digital picture, at least one region, at least one feature, or at least one pixel of the first stream of digital pictures. The substantial similarity may be achieved when a similarity between the at least one portion of the new stream of digital pictures and the at least one portion of the first stream of digital pictures exceeds a similarity threshold. The substantial similarity may be achieved when a number or a percentage of matching or partially matching digital pictures from the new stream of digital pictures and from the first stream of digital pictures exceeds a threshold number or threshold percentage. The substantial similarity may be achieved when a number or a percentage of matching or partially matching regions of at least one digital picture from the new stream of digital pictures and from the first stream of digital pictures exceeds a threshold number or threshold percentage. The substantial similarity may be achieved when a number or a percentage of matching or partially matching features of at least one digital picture from the new stream of digital pictures and from the first stream of digital pictures exceeds a threshold number or threshold percentage. The substantial similarity may be achieved when a number or a percentage of matching or partially matching pixels of at least one digital picture from the new stream of digital pictures and from the first stream of digital pictures exceeds a threshold number or threshold percentage. The substantial similarity may be achieved when one or more same or similar objects are recognized in the new stream of digital pictures and the first stream of digital pictures. In further embodiments, the determining that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes determining that a number or a percentage of matching digital pictures from the new stream of digital pictures and from the first stream of digital pictures exceeds a threshold number or threshold percentage. The matching digital pictures from the new stream of digital pictures and from the first stream of digital pictures may be determined factoring in at least one of: an order of a digital picture in a stream of digital pictures, an importance of a digital picture, a threshold for a similarity in a digital picture, or a threshold for a difference in a digital picture. In further embodiments, the determining that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes determining that a number or a percentage of matching regions from at least one digital picture of the new stream of digital pictures and from at least one digital picture of the first stream of digital pictures exceeds a threshold number or threshold percentage. The matching regions from at least one digital picture of the new stream of digital pictures and from at least one digital picture of the first stream of digital pictures may be determined factoring in at least one of: a location of a region, an importance of a region, a threshold for a similarity in a region, or a threshold for a difference in a region. In further embodiments, the determining that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes

determining that a number or a percentage of matching features from at least one digital picture of the new stream of digital pictures and from at least one digital picture of the first stream of digital pictures exceeds a threshold number or threshold percentage. The matching features from at least one digital picture of the new stream of digital pictures and from at least one digital picture of the first stream of digital pictures may be determined factoring in at least one of: a type of a feature, an importance of a feature, a location of a feature, a threshold for a similarity in a feature, or a threshold for a difference in a feature. In further embodiments, the determining that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes determining that a number or a percentage of matching pixels from at least one digital picture of the new stream of digital pictures and from at least one digital picture of the first stream of digital pictures exceeds a threshold number or threshold percentage. The matching pixels from at least one digital picture of the new stream of digital pictures and from at least one digital picture of the first stream of digital pictures may be determined factoring in at least one of: a location of a pixel, a threshold for a similarity in a pixel, or a threshold for a difference in a pixel. In further embodiments, the determining that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes recognizing a same person or object in the new and the first streams of digital pictures.

In certain embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures instead of or prior to an instruction set that would have been executed next. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying one or more instruction sets of the processor circuit. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes inserting the one or more instruction sets for operating the device correlated with the first stream of digital pictures into a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes redirecting the processor circuit to the one or more instruction sets for operating the device correlated with the first stream of digital pictures. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes redirecting the processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the one or more instruction sets for operating the device correlated with the first stream of digital pictures. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes transmitting, to the processor circuit for execution, the one or more instruction sets for operating the device correlated with the first stream of digital pictures. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes issuing an interrupt to the processor circuit and executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures following the interrupt. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying an element that is part of, operating on, or coupled to the processor circuit.

In some embodiments, the processor circuit includes a logic circuit, and wherein the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes causing the logic circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures. The logic circuit may include a microcontroller. The causing the logic circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures may include modifying an element of the logic circuit. The causing the logic circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures may include inserting the one or more instruction sets for operating the device correlated with the first stream of digital pictures into an element of the logic circuit. The causing the logic circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures may include redirecting the logic circuit to the one or more instruction sets for operating the device correlated with the first stream of digital pictures. The causing the logic circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures may include replacing inputs into the logic circuit with the one or more instruction sets for operating the device correlated with the first stream of digital pictures. The causing the logic circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures may include replacing outputs from the logic circuit with the one or more instruction sets for operating the device correlated with the first stream of digital pictures.

In certain embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes causing an application for operating the device to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures, the application running on the processor circuit.

In some embodiments, the system further comprises: an application including instruction sets for operating the device, the application running on the processor circuit, wherein the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying the application.

In certain embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes redirecting an application to the one or more instruction sets for operating the device correlated with the first stream of digital pictures, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of

digital pictures includes redirecting an application to one or more alternate instruction sets, the application running on the processor circuit, the alternate instruction sets comprising the one or more instruction sets for operating the device correlated with the first stream of digital pictures. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying one or more instruction sets of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying at least one of: the memory unit, a register of the processor circuit, a storage, or a repository where instruction sets are stored or used. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying one or more instruction sets for operating an application or an object of the application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying at least one of: an element of the processor circuit, an element of the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes utilizing an assembly language. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes adding or inserting additional code into a code of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes at least one of: modifying, removing, rewriting, or overwriting a code of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes at least one of: branching, redirecting, extending, or hot swapping a code of an application, the application running on the processor circuit. The branching or redirecting the code may include inserting at least one of: a branch, a jump, or a means for redirecting an execution.

In some embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes implementing a user's knowledge, style, or methodology of operating the device in a visual surrounding.

In certain embodiments, the system further comprises: an interface configured to cause execution of instruction sets, wherein the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures is caused by the interface. The interface may include a modification interface.

In some embodiments, the artificial intelligence unit is further configured to: receive at least one extra information. In further embodiments, the at least one extra information include one or more of: an information on a stream of digital pictures, an information on an object in the stream of digital pictures, an information on the device's visual surrounding, an information on an instruction set, an information on an application, an information on an object of the application, an information on the processor circuit, an information on the device, or an information on an user. In further embodiments, the artificial intelligence unit is further configured to: learn the first stream of digital pictures correlated with the at least one extra information. The learning the first stream of digital pictures correlated with at least one extra information may include correlating the first stream of digital pictures with the at least one extra information. The learning the first stream of digital pictures correlated with at least one extra

information may include storing the first stream of digital pictures correlated with the at least one extra information into the memory unit. In further embodiments, the anticipating the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes anticipating the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between an extra information correlated with the new stream of digital pictures and an extra information correlated with the first stream of digital pictures. The anticipating the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between an extra information correlated with the new stream of digital pictures and an extra information correlated with the first stream of digital pictures may include comparing an extra information correlated with the new stream of digital pictures and an extra information correlated with the first stream of digital pictures. The anticipating the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between an extra information correlated with the new stream of digital pictures and an extra information correlated with the first stream of digital pictures may include determining that a similarity between an extra information correlated with the new stream of digital pictures and an extra information correlated with the first stream of digital pictures exceeds a similarity threshold.

In certain embodiments, the artificial intelligence unit is further configured to: receive a second stream of digital pictures from the picture capturing apparatus; receive additional one or more instruction sets for operating the device from the processor circuit; and learn the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device. In further embodiments, the learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device and the learning the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device include creating a connection between the first stream of digital pictures correlated with the one or more instruction sets for operating the device and the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device. The connection includes or is associated with at least one of: an occurrence count, a weight, a parameter, or a data. In further embodiments, the learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device and the learning the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device include updating a connection between the first stream of digital pictures correlated with the one or more instruction sets for operating the device and the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device. The updating the connection between the first stream of digital pictures correlated with the one or more instruction sets for operating the device and the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device may include updating at least one of: an occurrence count, a weight, a parameter, or a data included in or associated with the connection. In further embodiments, the learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device includes storing the first stream of digital pictures correlated with the one or more instruction sets for operating the device into a first node of a data structure, and wherein the learning the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device includes storing the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device into a second node of the data structure. The data structure may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, or a knowledge structure. The learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device and the learning the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device may include creating a connection between the first node and the second node. The learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device and the learning the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device may include updating a connection between the first node and the second node. In further embodiments, the first stream of digital pictures correlated with the one or more instruction sets for operating the device is stored into a first node of a neural network and the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device is stored into a second node of the neural network. The first node and the second node may be connected by a connection. The first node may be part of a first layer of the neural network and the second node may be part of a second layer of the neural network. In further embodiments, the first stream of digital pictures correlated with the one or more instruction sets for operating the device is stored into a first node of a graph and the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device is stored into a second node of the graph. The first node and the second node may be connected by a connection. In further embodiments, the first stream of digital pictures correlated with the one or more instruction sets for operating the device is stored into a first node of a sequence and the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device is stored into a second node of the sequence.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first stream of digital pictures from a picture capturing apparatus. The operations may further include: receiving one or more instruction sets for operating a device. The operations may further include: learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device. The operations may further include: receiving a new stream of digital pictures from the picture capturing apparatus. The operations may further include: anticipating the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between the new stream of digital pictures and the first stream of digital pictures. The operations may further include: causing an execution of the one or more instruction sets for operating the device correlated with the first stream of digital pictures, the causing performed in response to the anticipating the one or more instruction sets for operating the device correlated

with the first stream of digital pictures based on at least a partial match between the new stream of digital pictures and the first stream of digital pictures, wherein the device performs one or more operations defined by the one or more instruction sets for operating the device correlated with the first stream of digital pictures, the one or more operations performed in response to the executing.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first stream of digital pictures from a picture capturing apparatus by one or more processor circuits. The method may further include: (b) receiving one or more instruction sets for operating a device by the one or more processor circuits. The method may further include: (c) learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device, the learning of (c) performed by the one or more processor circuits. The method may further include: (d) receiving a new stream of digital pictures from the picture capturing apparatus by the one or more processor circuits. The method may further include: (e) anticipating the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between the new stream of digital pictures and the first stream of digital pictures, the anticipating of (e) performed by the one or more processor circuits. The method may further include: (f) executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures, the executing of (f) performed in response to the anticipating of (e). The method may further include: (g) performing, by the device, one or more operations defined by the one or more instruction sets for operating the device correlated with the first stream of digital pictures, the one or more operations performed in response to the executing of (f).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems as applicable as well as the following embodiments.

In some embodiments, the first stream of digital pictures correlated with the one or more instruction sets for operating the device includes a unit of knowledge of how the device operated in a visual surrounding. In further embodiments, the first stream of digital pictures correlated with the one or more instruction sets for operating the device is included in a neuron, a node, a vertex, or an element of a data structure. The data structure may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, or a knowledge structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the first stream of digital pictures correlated with the one or more instruction sets for operating the device is structured into a knowledge cell. In further embodiments, the knowledge cell includes a unit of knowledge of how the device operated in a visual surrounding. In further embodiments, the knowledge cell is included in a neuron, a node, a vertex, or an element of a data structure. The data structure may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, or a knowledge structure. Some of the neurons, nodes, vertices, or elements may be interconnected.

In certain embodiments, the learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device includes correlating the first stream of digital pictures with the one or more instruction sets for operating the device. The correlating the first stream of digital pictures with the one or more instruction sets for operating the device may include generating a knowledge cell, the knowledge cell comprising the first stream of digital pictures correlated with the one or more instruction sets for operating the device. The correlating the first stream of digital pictures with the one or more instruction sets for operating the device may include structuring a unit of knowledge of how the device operated in a visual surrounding. In further embodiments, the learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device includes learning a user's knowledge, style, or methodology of operating the device in a visual surrounding. In further embodiments, the learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device includes spontaneous learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device.

In some embodiments, the learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device includes storing, into a memory unit, the first stream of digital pictures correlated with the one or more instruction sets for operating the device, the first stream of digital pictures correlated with the one or more instruction sets for operating the device being part of a stored plurality of streams of digital pictures correlated with one or more instruction sets for operating the device. In further embodiments, the plurality of streams of digital pictures correlated with one or more instruction sets for operating the device include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, the plurality of streams of digital pictures correlated with one or more instruction sets for operating the device are organized into a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, each of the plurality of streams of digital pictures correlated with one or more instruction sets for operating the device is included in a neuron, a node, a vertex, or an element of a data structure. The data structure may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, or a knowledge structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the plurality of streams of digital pictures correlated with one or more instruction sets for operating the device include a user's knowledge, style, or methodology of operating the device in visual surroundings. In further embodiments, the plurality of streams of digital pictures correlated with one or more instruction sets for operating the device are stored on a remote computing device. In further embodiments, the plurality of streams of digital pictures correlated with one or more instruction sets for operating the device include an artificial intelligence system for knowledge structuring, storing, or representation. The artificial intelligence system for knowledge structuring, storing, or representation may include at least one of: a deep learning system, a supervised learning system, an unsupervised learning system, a neural network, a search-based system, an optimization-based system, a logic-based system, a fuzzy logic-based system, a tree-based system, a graph-based system, a hierarchical system, a symbolic system, a sub-symbolic system, an evolutionary system, a genetic system,

a multi-agent system, a deterministic system, a probabilistic system, or a statistical system.

In further embodiments, the anticipating the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes comparing at least one portion of the new stream of digital pictures with at least one portion of the first stream of digital pictures. The at least one portion of the new stream of digital pictures may include at least one digital picture, at least one region, at least one feature, or at least one pixel of the new stream of digital pictures. The at least one portion of the first stream of digital pictures may include at least one digital picture, at least one region, at least one feature, or at least one pixel of the first stream of digital pictures. The comparing the at least one portion of the new stream of digital pictures with the at least one portion of the first stream of digital pictures may include comparing at least one digital picture of the new stream of digital pictures with at least one digital picture of the first stream of digital pictures. The comparing the at least one portion of the new stream of digital pictures with the at least one portion of the first stream of digital pictures may include comparing at least one region of at least one digital picture of the new stream of digital pictures with at least one region of at least one digital picture of the first stream of digital pictures. The comparing the at least one portion of the new stream of digital pictures with the at least one portion of the first stream of digital pictures may include comparing at least one feature of at least one digital picture of the new stream of digital pictures with at least one feature of at least one digital picture of the first stream of digital pictures. The comparing the at least one portion of the new stream of digital pictures with the at least one portion of the first stream of digital pictures may include comparing at least one pixel of at least one digital picture of the new stream of digital pictures with at least one pixel of at least one digital picture of the first stream of digital pictures. The comparing the at least one portion of the new stream of digital pictures with the at least one portion of the first stream of digital pictures may include at least one of: performing a color adjustment, performing a size adjustment, performing a content manipulation, performing temporal alignment, performing dynamic time warping, utilizing a transparency, or utilizing a mask on the new or the first stream of digital pictures. The comparing the at least one portion of the new stream of digital pictures with the at least one portion of the first stream of digital pictures may include recognizing at least one person or object in the new stream of digital pictures and at least one person or object in the first stream of digital pictures, and comparing the at least one person or object from the new stream of digital pictures with the at least one person or object from the first stream of digital pictures.

In some embodiments, the anticipating the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes determining that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures. In further embodiments, the determining that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes determining that there is at least a partial match between one or more portions of the new stream of digital pictures and one or more portions of the first stream of digital pictures. In further embodiments, the determining that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes determining that a similarity between at least one portion of the new stream of digital pictures and at least one portion of the first stream of digital pictures exceeds a similarity threshold. In further embodiments, the determining that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes determining a substantial similarity between at least one portion of the new stream of digital pictures and at least one portion of the first stream of digital pictures. The at least one portion of the new stream of digital pictures may include at least one digital picture, at least one region, at least one feature, or at least one pixel of the new stream of digital pictures. The at least one portion of the first stream of digital pictures may include at least one digital picture, at least one region, at least one feature, or at least one pixel of the first stream of digital pictures. The substantial similarity may be achieved when a similarity between the at least one portion of the new stream of digital pictures and the at least one portion of the first stream of digital pictures exceeds a similarity threshold. The substantial similarity may be achieved when a number or a percentage of matching or partially matching digital pictures from the new stream of digital pictures and from the first stream of digital pictures exceeds a threshold number or threshold percentage. The substantial similarity may be achieved when a number or a percentage of matching or partially matching regions of at least one digital picture from the new stream of digital pictures and from the first stream of digital pictures exceeds a threshold number or threshold percentage. The substantial similarity may be achieved when a number or a percentage of matching or partially matching features of at least one digital picture from the new stream of digital pictures and from the first stream of digital pictures exceeds a threshold number or threshold percentage. The substantial similarity may be achieved when a number or a percentage of matching or partially matching pixels of at least one digital picture from the new stream of digital pictures and from the first stream of digital pictures exceeds a threshold number or threshold percentage. The substantial similarity may be achieved when one or more same or similar objects are recognized in the new stream of digital pictures and the first stream of digital pictures. In further embodiments, the determining that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes determining that a number or a percentage of matching digital pictures from the new stream of digital pictures and from the first stream of digital pictures exceeds a threshold number or threshold percentage. The matching digital pictures from the new stream of digital pictures and from the first stream of digital pictures may be determined factoring in at least one of: an order of a digital picture in a stream of digital pictures, an importance of a digital picture, a threshold for a similarity in a digital picture, or a threshold for a difference in a digital picture. In further embodiments, the determining that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes determining that a number or a percentage of matching regions from at least one digital picture of the new stream of digital pictures and from at least one digital picture of the first stream of digital pictures exceeds a threshold number or threshold percentage. The matching regions from at least one digital picture of the new stream of digital pictures and from at least one digital picture of the first stream of digital pictures may be determined factoring in at least one of: a

location of a region, an importance of a region, a threshold for a similarity in a region, or a threshold for a difference in a region. In further embodiments, the determining that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes determining that a number or a percentage of matching features from at least one digital picture of the new stream of digital pictures and from at least one digital picture of the first stream of digital pictures exceeds a threshold number or threshold percentage. The matching features from at least one digital picture of the new stream of digital pictures and from at least one digital picture of the first stream of digital pictures may be determined factoring in at least one of: a type of a feature, an importance of a feature, a location of a feature, a threshold for a similarity in a feature, or a threshold for a difference in a feature. In further embodiments, the determining that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes determining that a number or a percentage of matching pixels from at least one digital picture of the new stream of digital pictures and from at least one digital picture of the first stream of digital pictures exceeds a threshold number or threshold percentage. The matching pixels from at least one digital picture of the new stream of digital pictures and from at least one digital picture of the first stream of digital pictures may be determined factoring in at least one of: a location of a pixel, a threshold for a similarity in a pixel, or a threshold for a difference in a pixel. In further embodiments, the determining that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes recognizing a same person or object in the new and the first streams of digital pictures.

In certain embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures instead of or prior to an instruction set that would have been executed next. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying one or more instruction sets. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying a register or an element of a processor circuit. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes inserting the one or more instruction sets for operating the device correlated with the first stream of digital pictures into a register or an element of a processor circuit. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes redirecting a processor circuit to the one or more instruction sets for operating the device correlated with the first stream of digital pictures. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes redirecting a processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the one or more instruction sets for operating the device correlated with the first stream of digital pictures. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes transmitting, to a processor circuit for execution, the one or more instruction sets for operating the device correlated with the first stream of digital pictures. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes issuing an interrupt to a processor circuit and executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures following the interrupt. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying an element that is part of, operating on, or coupled to a processor circuit.

In some embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes executing, by a logic circuit, the one or more instruction sets for operating the device correlated with the first stream of digital pictures. The logic circuit may include a microcontroller. The executing, by the logic circuit, the one or more instruction sets for operating the device correlated with the first stream of digital pictures may include modifying an element of the logic circuit. The executing, by the logic circuit, the one or more instruction sets for operating the device correlated with the first stream of digital pictures may include inserting the one or more instruction sets for operating the device correlated with the first stream of digital pictures into an element of the logic circuit. The executing, by the logic circuit, the one or more instruction sets for operating the device correlated with the first stream of digital pictures may include redirecting the logic circuit to the one or more instruction sets for operating the device correlated with the first stream of digital pictures. The executing, by the logic circuit, the one or more instruction sets for operating the device correlated with the first stream of digital pictures may include replacing inputs into the logic circuit with the one or more instruction sets for operating the device correlated with the first stream of digital pictures. The executing, by the logic circuit, the one or more instruction sets for operating the device correlated with the first stream of digital pictures my include replacing outputs from the logic circuit with the one or more instruction sets for operating the device correlated with the first stream of digital pictures.

In certain embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes executing, by an application for operating the device, the one or more instruction sets for operating the device correlated with the first stream of digital pictures. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying an application, the application including instruction sets for operating the device. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes redirecting an application to the one or more instruction sets for operating the device correlated with the first stream of digital pictures. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes redirecting an application to one or more alternate instruction sets, the alternate instruction sets comprising the one or more instruction sets for operating the device correlated with the first stream of digital pictures. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying one or more instruction sets of an application. In further embodiments, the executing the one or more instruction sets for operating

the device correlated with the first stream of digital pictures includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying at least one of: a memory unit, a register of a processor circuit, a storage, or a repository where instruction sets are stored or used. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying one or more instruction sets for operating an application or an object of the application. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying at least one of: an element of a processor circuit, an element of the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of an application. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes utilizing an assembly language. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes adding or inserting additional code into a code of an application. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes at least one of: modifying, removing, rewriting, or overwriting a code of an application. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes at least one of: branching, redirecting, extending, or hot swapping a code of an application. The branching or redirecting the code may include inserting at least one of: a branch, a jump, or a means for redirecting an execution. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes implementing a user's knowledge, style, or methodology of operating the device in a visual surrounding. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures via an interface. The interface may include a modification interface.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving at least one extra information. In further embodiments, the at least one extra information include one or more of: an information on a stream of digital pictures, an information on an object in the stream of digital pictures, an information on the device's visual surrounding, an information on an instruction set, an information on an application, an information on an object of the application, an information on a processor circuit, an information on the device, or an information on an user. In further embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: learning the first stream of digital pictures correlated with the at least one extra information. The learning the first stream of digital pictures correlated with at least one extra information may include correlating the first stream of digital pictures with the at least one extra information. The learning the first stream of digital pictures correlated with at least one extra information may include storing the first stream of digital pictures correlated with the at least one extra information into a memory unit. In further embodiments, the anticipating the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes anticipating the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between an extra information correlated with the new stream of digital pictures and an extra information correlated with the first stream of digital pictures. The anticipating the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between an extra information correlated with the new stream of digital pictures and an extra information correlated with the first stream of digital pictures may include comparing an extra information correlated with the new stream of digital pictures and an extra information correlated with the first stream of digital pictures. The anticipating the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between an extra information correlated with the new stream of digital pictures and an extra information correlated with the first

stream of digital pictures may include determining that a similarity between an extra information correlated with the new stream of digital pictures and an extra information correlated with the first stream of digital pictures exceeds a similarity threshold.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving a second stream of digital pictures from the picture capturing apparatus; receiving additional one or more instruction sets for operating the device; and learning the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device. In further embodiments, the learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device and the learning the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device include creating a connection between the first stream of digital pictures correlated with the one or more instruction sets for operating the device and the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device. The connection may include or be associated with at least one of: an occurrence count, a weight, a parameter, or a data. In further embodiments, the learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device and the learning the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device include updating a connection between the first stream of digital pictures correlated with the one or more instruction sets for operating the device and the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device. The updating the connection between the first stream of digital pictures correlated with the one or more instruction sets for operating the device and the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device may include updating at least one of: an occurrence count, a weight, a parameter, or a data included in or associated with the connection. In further embodiments, the learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device includes storing the first stream of digital pictures correlated with the one or more instruction sets for operating the device into a first node of a data structure, and wherein the learning the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device includes storing the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device into a second node of the data structure. The data structure may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, or a knowledge structure. The learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device and the learning the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device may include creating a connection between the first node and the second node. The learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device and the learning the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device may include updating a connection between the first node and the second node. In further embodiments, the first stream of digital pictures correlated with the one or more instruction sets for operating the device is stored into a first node of a neural network and the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device is stored into a second node of the neural network. The first node and the second node may be connected by a connection. The first node may be part of a first layer of the neural network and the second node may be part of a second layer of the neural network. In further embodiments, the first stream of digital pictures correlated with the one or more instruction sets for operating the device is stored into a first node of a graph and the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device is stored into a second node of the graph. The first node and the second node may be connected by a connection. In further embodiments, the first stream of digital pictures correlated with the one or more instruction sets for operating the device is stored into a first node of a sequence and the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device is stored into a second node of the sequence.

In some aspects, the disclosure relates to a system for learning a visual surrounding for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises a processor circuit configured to execute instruction sets for operating a device. The system may further include a memory unit configured to store data. The system may further include a picture capturing apparatus configured to capture digital pictures. The system may further include an artificial intelligence unit. In some embodiments, the artificial intelligence unit may be configured to: receive a first stream of digital pictures from the picture capturing apparatus. The artificial intelligence unit may be further configured to: receive one or more instruction sets for operating the device from the processor circuit. The artificial intelligence unit may be further configured to: learn the first stream of digital pictures correlated with the one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first stream of digital pictures from a picture capturing apparatus. The operations may further include: receiving one or more instruction sets for operating a device. The operations may further include: learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first stream of digital pictures from a picture capturing apparatus by one or more processor circuits. The method may further include: (b) receiving one or more instruction sets for operating a device by the one or more processor circuits. The method may further include: (c) learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device, the learning of (c) performed by the one or more processor circuits.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems as applicable.

In some aspects, the disclosure relates to a system for using a visual surrounding for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises a processor circuit configured to execute instruction sets for operating a device. The system may further include a memory unit configured to store data. The system may further include a picture capturing apparatus configured to capture digital pictures. The system may further include an artificial intelligence unit. In some embodiments, the artificial intelligence unit may be configured to: access the memory unit that stores a plurality of streams of digital pictures correlated with one or more instruction sets for operating the device, the plurality including a first stream of digital pictures correlated with one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: receive a new stream of digital pictures from the picture capturing apparatus. The artificial intelligence unit may be further configured to: anticipate the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between the new stream of digital pictures and the first stream of digital pictures. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures, the executing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the one or more instruction sets for operating the device correlated with the first stream of digital pictures, the one or more operations performed in response to the executing by the processor circuit.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: accessing a memory unit that stores a plurality of streams of digital pictures correlated with one or more instruction sets for operating a device, the plurality including a first stream of digital pictures correlated with one or more instruction sets for operating the device. The operations may further include: receiving a new stream of digital pictures from a picture capturing apparatus. The operations may further include: anticipating the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between the new stream of digital pictures and the first stream of digital pictures. The operations may further include: causing an execution of the one or more instruction sets for operating the device correlated with the first stream of digital pictures, the causing performed in response to the anticipating the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between the new stream of digital pictures and the first stream of digital pictures, wherein the device performs one or more operations defined by the one or more instruction sets for operating the device correlated with the first stream of digital pictures, the one or more operations performed in response to the executing.

In some aspects, the disclosure relates to a method comprising: (a) accessing a memory unit that stores a plurality of streams of digital pictures correlated with one or more instruction sets for operating a device, the plurality including a first stream of digital pictures correlated with one or more instruction sets for operating the device, the accessing of (a) performed by the one or more processor circuits. The method may further include: (b) receiving a new stream of digital pictures from a picture capturing apparatus by the one or more processor circuits. The method may further include: (c) anticipating the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between the new stream of digital pictures and the first stream of digital pictures, the anticipating of (c) performed by the one or more processor circuits. The method may further include: (d) executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures, the executing of (d) performed in response to the anticipating of (c). The method may further include: (e) performing, by the device, one or more operations defined by the one or more instruction sets for operating the device correlated with the first stream of digital pictures, the one or more operations performed in response to the executing of (d).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems as applicable.

In some aspects, the disclosure relates to a system for learning and using a visual surrounding for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises a logic circuit configured to receive inputs and produce outputs, the outputs for operating a device. The system may further include a memory unit configured to store data. The system may further include a picture capturing apparatus configured to capture digital pictures. The system may further include an artificial intelligence unit. In some embodiments, the artificial intelligence unit may be configured to: receive a first digital picture from the picture capturing apparatus. The artificial intelligence unit may be further configured to: receive at least one input, wherein the at least one input is also received by the logic circuit. The artificial intelligence unit may be further configured to: learn the first digital picture correlated with the at least one input. The artificial intelligence unit may be further configured to: receive a new digital picture from the picture capturing apparatus. The artificial intelligence unit may be further configured to: anticipate the at least one input correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture. The artificial intelligence unit may be further configured to: cause the logic circuit to receive the at least one input correlated with the first digital picture, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs at least one operation defined by at least one output for operating the device produced by the logic circuit.

In certain embodiments, the logic circuit configured to receive inputs and produce outputs includes a logic circuit configured to produce outputs based at least in part on logic operations performed on the inputs. In further embodiments, the learning the first digital picture correlated with the at least one input includes correlating the first digital picture with the at least one input. In further embodiments, the learning the first digital picture correlated with the at least one input includes storing, into the memory unit, the first digital picture correlated with the at least one input, the first digital picture correlated with the at least input being part of a stored plurality of digital pictures correlated with at least

one input. In further embodiments, the anticipating the at least one input correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture includes comparing at least one portion of the new digital picture with at least one portion of the first digital picture. In further embodiments, the anticipating the at least one input correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture includes determining that there is at least a partial match between the new digital picture and the first digital picture. In further embodiments, the causing the logic circuit to receive the at least one input correlated with the first digital picture includes transmitting, to the logic circuit, the at least one input correlated with the first digital picture. In further embodiments, the causing the logic circuit to receive the at least one input correlated with the first digital picture includes replacing at least one input into the logic circuit with the at least one input correlated with the first digital picture.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first digital picture from a picture capturing apparatus. The operations may further include: receiving at least one input, wherein the at least one input is also received by a logic circuit, and wherein the logic circuit is configured to receive inputs and produce outputs, the outputs for operating a device. The operations may further include: learning the first digital picture correlated with the at least one input. The operations may further include: receiving a new digital picture from the picture capturing apparatus. The operations may further include: anticipating the at least one input correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture. The operations may further include: causing the logic circuit to receive the at least one input correlated with the first digital picture, the causing performed in response to the anticipating the at least one input correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture, wherein the device performs at least one operation defined by at least one output for operating the device produced by the logic circuit.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first digital picture from a picture capturing apparatus by one or more processor circuits. The method may further include: (b) receiving at least one input by the one or more processor circuits, wherein the at least one input are also received by a logic circuit, and wherein the logic circuit is configured to receive inputs and produce outputs, the outputs for operating a device. The method may further include: (c) learning the first digital picture correlated with the at least one input, the learning of (c) performed by the one or more processor circuits. The method may further include: (d) receiving a new digital picture from the picture capturing apparatus by the one or more processor circuits. The method may further include: (e) anticipating the at least one input correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture, the anticipating of (e) performed by the one or more processor circuits. The method may further include: (f) receiving, by the logic circuit, the at least one input correlated with the first digital picture, the receiving of (f) performed in response to the anticipating of (e). The method may further include: (g) performing, by the device, at least one operation defined by at least one output for operating the device produced by the logic circuit.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems as applicable.

In some aspects, the disclosure relates to a system for learning and using a visual surrounding for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises a logic circuit configured to receive inputs and produce outputs, the outputs for operating a device. The system may further include a memory unit configured to store data. The system may further include a picture capturing apparatus configured to capture digital pictures. The system may further include an artificial intelligence unit. In some embodiments, the artificial intelligence unit may be configured to: receive a first digital picture from the picture capturing apparatus. The artificial intelligence unit may be further configured to: receive at least one output, the at least one output transmitted from the logic circuit. The artificial intelligence unit may be further configured to: learn the first digital picture correlated with the at least one output. The artificial intelligence unit may be further configured to: receive a new digital picture from the picture capturing apparatus. The artificial intelligence unit may be further configured to: anticipate the at least one output correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture. The artificial intelligence unit may be further configured to: cause the device to perform at least one operation defined by the at least one output correlated with the first digital picture.

In certain embodiments, the logic circuit configured to receive inputs and produce outputs includes a logic circuit configured to produce outputs based at least in part on logic operations performed on the inputs. In further embodiments, the learning the first digital picture correlated with the at least one output includes correlating the first digital picture with the at least one output. In further embodiments, the learning the first digital picture correlated with the at least one output includes storing, into the memory unit, the first digital picture correlated with the at least one output, the first digital picture correlated with the at least output being part of a stored plurality of digital pictures correlated with at least one output. In further embodiments, the anticipating the at least one output correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture includes comparing at least one portion of the new digital picture with at least one portion of the first digital picture. In further embodiments, the anticipating the at least one output correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture includes determining that there is at least a partial match between the new digital picture and the first digital picture. In further embodiments, the causing the device to perform at least one operation defined by the at least one output correlated with the first digital picture includes replacing at least one output from the logic circuit with the at least one output correlated with the first digital picture.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when

executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first digital picture from the picture capturing apparatus. The operations may further include: receiving at least one output, the at least one output transmitted from a logic circuit, wherein the logic circuit is configured to receive inputs and produce outputs, the outputs for operating a device. The operations may further include: learning the first digital picture correlated with the at least one output. The operations may further include: receiving a new digital picture from the picture capturing apparatus. The operations may further include: anticipating the at least one output correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture. The operations may further include: causing the device to perform at least one operation defined by the at least one output correlated with the first digital picture.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first digital picture from the picture capturing apparatus by one or more processor circuits. The method may further include: (b) receiving at least one output by the one or more processor circuits, the at least one output transmitted from a logic circuit, wherein the logic circuit is configured to receive inputs and produce outputs, the outputs for operating a device. The method may further include: (c) learning the first digital picture correlated with the at least one output, the learning of (c) performed by the one or more processor circuits. The method may further include: (d) receiving a new digital picture from the picture capturing apparatus by the one or more processor circuits. The method may further include: (e) anticipating the at least one output correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture, the anticipating of (e) performed by the one or more processor circuits. The method may further include: (f) performing, by the device, at least one operation defined by the at least one output correlated with the first digital picture.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems as applicable.

In some aspects, the disclosure relates to a system for learning and using a visual surrounding for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises an actuator configured to receive inputs and perform motions. The system may further include a memory unit configured to store data. The system may further include a picture capturing apparatus configured to capture digital pictures. The system may further include an artificial intelligence unit. In some embodiments, the artificial intelligence unit may be configured to: receive a first digital picture from the picture capturing apparatus. The artificial intelligence unit may be further configured to: receive at least one input, wherein the at least one input is also received by the actuator. The artificial intelligence unit may be further configured to: learn the first digital picture correlated with the at least one input. The artificial intelligence unit may be further configured to: receive a new digital picture from the picture capturing apparatus. The artificial intelligence unit may be further configured to. The artificial intelligence unit may be further configured to: anticipate the at least one input correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture. The artificial intelligence unit may be further configured to: cause the actuator to receive the at least one input correlated with the first digital picture, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the actuator performs at least one motion defined by the at least one input correlated with the first digital picture.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first digital picture from a picture capturing apparatus. The operations may further include: receiving at least one input, wherein the at least one input is also received by an actuator, and wherein the actuator is configured to receive inputs and perform motions. The operations may further include: learning the first digital picture correlated with the at least one input. The operations may further include: receiving a new digital picture from the picture capturing apparatus. The operations may further include: anticipating the at least one input correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture. The operations may further include: causing the actuator to receive the at least one input correlated with the first digital picture, the causing performed in response to the anticipating the at least one input correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture, wherein the actuator performs at least one motion defined by the at least one input correlated with the first digital picture.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first digital picture from a picture capturing apparatus by one or more processor circuits. The method may further include: (b) receiving at least one input by the one or more processor circuits, wherein the at least one input are also received by an actuator, and wherein the actuator is configured to receive inputs and perform motions. The method may further include: (c) learning the first digital picture correlated with the at least one input, the learning of (c) performed by the one or more processor circuits. The method may further include: (d) receiving a new digital picture from the picture capturing apparatus by the one or more processor circuits. The method may further include: (e) anticipating the at least one input correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture, the anticipating of (e) performed by the one or more processor circuits. The method may further include: (f) receiving, by the actuator, the at least one input correlated with the first digital picture, the receiving of (f) performed in response to the anticipating of (e). The method may further include: (g) performing, by the actuator, at least one motion defined by the at least one input correlated with the first digital picture.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems as applicable.

Other features and advantages of the disclosure will become apparent from the following description, including the claims and drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** illustrates a block diagram of Computing Device **70** that can provide processing capabilities used in some of the disclosed embodiments.

FIG. **2** illustrates an embodiment of Device **98** comprising Unit for Learning and/or Using Visual Surrounding for Autonomous Device Operation (VSADO Unit **100**).

FIG. **3** illustrates some embodiments of obtaining instruction sets, data, and/or other information through tracing, profiling, or sampling of Processor **11** registers, memory, or other computing system components.

FIGS. **4**A-**4**B illustrate some embodiments of obtaining instruction sets, data, and/or other information through tracing, profiling, or sampling of Logic Circuit **250**.

FIGS. **5**A-**5**E illustrate some embodiments of Instruction Sets **526**.

FIGS. **6**A-**6**B illustrate some embodiments of Extra Information **527**.

FIG. **7** illustrates an embodiment where VSADO Unit **100** is part of or operating on Processor **11**.

FIG. **8** illustrates an embodiment where VSADO Unit **100** resides on Server **96** accessible over Network **95**.

FIG. **9** illustrates an embodiment where Picture Capturing Apparatus **90** is part of Remote Device **97** accessible over Network **95**.

FIG. **10** illustrates an embodiment of VSADO Unit **100** comprising Picture Recognizer **350**.

FIG. **11** illustrates an embodiment of Artificial Intelligence Unit **110**.

FIG. **12** illustrates an embodiment of Knowledge Structuring Unit **520** correlating individual Digital Pictures **525** with any Instruction Sets **526** and/or Extra Info **527**.

FIG. **13** illustrates another embodiment of Knowledge Structuring Unit **520** correlating individual Digital Pictures **525** with any Instruction Sets **526** and/or Extra Info **527**.

FIG. **14** illustrates an embodiment of Knowledge Structuring Unit **520** correlating streams of Digital Pictures **525** with any Instruction Sets **526** and/or Extra Info **527**.

FIG. **15** illustrates another embodiment of Knowledge Structuring Unit **520** correlating streams of Digital Pictures **525** with any Instruction Sets **526** and/or Extra Info **527**.

FIG. **16** illustrates various artificial intelligence methods, systems, and/or models that can be utilized in VSADO Unit **100** embodiments.

FIGS. **17**A-**17**C illustrate embodiments of interconnected Knowledge Cells **800** and updating weights of Connections **853**.

FIG. **18** illustrates an embodiment of learning Knowledge Cells **800** comprising one or more Digital Pictures **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Collection of Knowledge Cells **530***d*.

FIG. **19** illustrates an embodiment of learning Knowledge Cells **800** comprising one or more Digital Pictures **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Neural Network **530***a*.

FIG. **20** illustrates an embodiment of learning Knowledge Cells **800** comprising one or more Digital Pictures **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Neural Network **530***a* comprising shortcut Connections **853**.

FIG. **21** illustrates an embodiment of learning Knowledge Cells **800** comprising one or more Digital Pictures **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Graph **530***b*.

FIG. **22** illustrates an embodiment of learning Knowledge Cells **800** comprising one or more Digital Pictures **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Collection of Sequences **530***c*.

FIG. **23** illustrates an embodiment of determining anticipatory Instruction Sets **526** from a single Knowledge Cell **800**.

FIG. **24** illustrates an embodiment of determining anticipatory Instruction Sets **526** by traversing a single Knowledge Cell **800**.

FIG. **25** illustrates an embodiment of determining anticipatory Instruction Sets **526** using collective similarity comparisons.

FIG. **26** illustrates an embodiment of determining anticipatory Instruction Sets **526** using Neural Network **530***a*.

FIG. **27** illustrates an embodiment of determining anticipatory Instruction Sets **526** using Graph **530***b*.

FIG. **28** illustrates an embodiment of determining anticipatory Instruction Sets **526** using Collection of Sequences **530***c*.

FIG. **29** illustrates some embodiments of modifying execution and/or functionality of Processor **11** through modification of Processor **11** registers, memory, or other computing system components.

FIGS. **30**A-**30**B illustrate some embodiments of modifying execution and/or functionality of Logic Circuit **250** through modification of inputs and/or outputs of Logic Circuit **250**.

FIG. **31** illustrates a flow chart diagram of an embodiment of method **6100** for learning and/or using visual surrounding for autonomous device operation.

FIG. **32** illustrates a flow chart diagram of an embodiment of method **6200** for learning and/or using visual surrounding for autonomous device operation.

FIG. **33** illustrates a flow chart diagram of an embodiment of method **6300** for learning and/or using visual surrounding for autonomous device operation.

FIG. **34** illustrates a flow chart diagram of an embodiment of method **6400** for learning and/or using visual surrounding for autonomous device operation.

FIG. **35** illustrates a flow chart diagram of an embodiment of method **6500** for learning and/or using visual surrounding for autonomous device operation.

FIG. **36** illustrates a flow chart diagram of an embodiment of method **6600** for learning and/or using visual surrounding for autonomous device operation.

FIG. **37** illustrates an exemplary embodiment of Computing-enabled Machine **98***a*.

FIG. **38** illustrates an exemplary embodiment of Computing-enabled Machine **98***a* comprising or coupled to a plurality of Picture Capturing Apparatuses **90**.

FIG. **39** illustrates an exemplary embodiment of Fixture **98***b*.

FIG. **40** illustrates an exemplary embodiment of Control Device **98***c*.

FIG. **41** illustrates an exemplary embodiment of Smartphone **98***d*.

Like reference numerals in different figures indicate like elements. Horizontal or vertical " . . . " or other such indicia may be used to indicate additional instances of the same type of element n, m, x, or other such letters or indicia represent integers or other sequential numbers that follow the sequence where they are indicated. It should be noted that n, m, x, or other such letters or indicia may represent different numbers in different elements even where the elements are depicted in the same figure. In general, n, m, x, or other such letters or indicia may follow the sequence and/or context where they are indicated. Any of these or other such letters or indicia may be used interchangeably depending on the

context and space available. The drawings are not necessarily to scale, with emphasis instead being placed upon illustrating the embodiments, principles, and concepts of the disclosure. A line or arrow between any of the disclosed elements comprises an interface that enables the coupling, connection, and/or interaction between the elements.

DETAILED DESCRIPTION

The disclosed artificially intelligent devices, systems, and methods for learning and/or using visual surrounding for autonomous device operation comprise apparatuses, systems, methods, features, functionalities, and/or applications that enable learning one or more digital pictures of a device's surrounding along with correlated instruction sets for operating the device, storing this knowledge in a knowledgebase (i.e. neural network, graph, sequences, etc.), and autonomously operating a device. The disclosed artificially intelligent devices, systems, and methods for learning and/or using visual surrounding for autonomous device operation, any of their elements, any of their embodiments, or a combination thereof can generally be referred to as VSADO, VSADO Unit, or as other similar name or reference.

Referring now to FIG. **1**, an embodiment is illustrated of Computing Device **70** (also referred to simply as computing device or other similar name or reference, etc.) that can provide processing capabilities used in some embodiments of the forthcoming disclosure. Later described devices and systems, in combination with processing capabilities of Computing Device **70**, enable learning and/or using a device's visual surrounding for autonomous device operation and/or other functionalities described herein. Various embodiments of the disclosed devices, systems, and/or methods include hardware, functions, logic, programs, and/or a combination thereof that can be provided or implemented on any type or form of computing, computing enabled, or other device such as a mobile device, a computer, a computing enabled telephone, a server, a cloud device, a gaming device, a television device, a digital camera, a GPS receiver, a media player, an embedded device, a supercomputer, a wearable device, an implantable device, or any other type or form of computing, computing enabled, or other device capable of performing the operations described herein.

In some designs, Computing Device **70** comprises hardware, processing techniques or capabilities, programs, or a combination thereof. Computing Device **70** includes one or more central processing units, which may also be referred to as processors **11**. Processor **11** includes one or more memory ports **10** and/or one or more input-output ports, also referred to as I/O ports **15**, such as I/O ports **15**A and **15**B. Processor **11** may be special or general purpose. Computing Device **70** may further include memory **12**, which can be connected to the remainder of the components of Computing Device **70** via bus **5**. Memory **12** can be connected to processor **11** via memory port **10**. Computing Device **70** may also include display device **21** such as a monitor, projector, glasses, and/or other display device. Computing Device **70** may also include Human-machine Interface **23** such as a keyboard, a pointing device, a mouse, a touchscreen, a joystick, and/or other input device that can be connected with the remainder of the Computing Device **70** components via I/O control **22**. In some implementations, Human-machine Interface **23** can be connected with bus **5** or directly connected with specific components of Computing Device **70**. Computing Device **70** may include additional elements, such as one or more input/output devices **13**. Processor **11** may include or be interfaced with cache memory **14**. Storage **27** may include memory, which provides an operating system, also referred to as OS **17**, additional application programs **18** operating on OS **17**, and/or data space **19** in which additional data or information can be stored. Alternative memory device **16** can be connected to the remaining components of Computing Device **70** via bus **5**. Network interface **25** can also be connected with bus **5** and be used to communicate with external computing devices via a network. Some or all described elements of Computing Device **70** can be directly or operatively connected or coupled with each other using any other connection means known in art. Other additional elements may be included as needed, or some of the disclosed ones may be excluded, or a combination thereof may be utilized in alternate implementations of Computing Device **70**.

Processor **11** includes any logic circuitry that can respond to or process instructions fetched from memory **12** or other element. Processor **11** may also include any combination of hardware and/or processing techniques or capabilities for implementing or executing logic functions or programs. Processor **11** may include a single core or a multi core processor. Processor **11** includes the functionality for loading operating system **17** and operating any application programs **18** thereon. In some embodiments, Processor **11** can be provided in a microprocessing or a processing unit, such as, for example, Snapdragon processor produced by Qualcomm Inc., processor by Intel Corporation of Mountain View, Calif., processor manufactured by Motorola Corporation of Schaumburg, Ill.; processor manufactured by Transmeta Corporation of Santa Clara, Calif.; the RS/6000 processor, processor manufactured by International Business Machines of White Plains, N.Y.; processor manufactured by Advanced Micro Devices of Sunnyvale, Calif., or any computing unit for performing similar functions. In other embodiments, processor **11** can be provided in a graphics processing unit (GPU), visual processing unit (VPU), or other highly parallel processing unit or circuit such as, for example, nVidia GeForce line of GPUs, AMD Radeon line of GPUs, and/or others. Such GPUs or other highly parallel processing units may provide superior performance in processing operations on neural networks and/or other data structures. In further embodiments, processor **11** can be provided in a micro controller such as, for example, Texas instruments, Atmel, Microchip Technology, ARM, Silicon Labs, Intel, and/or other lines of micro controllers, and/or others. In further embodiments, processor **11** includes any circuit (i.e. logic circuit, etc.) or device for performing logic operations. Computing Device **70** can be based on one or more of the aforementioned or other processors capable of operating as described herein.

Memory **12** includes one or more memory chips capable of storing data and allowing any storage location to be accessed by processor **11** and/or other element. Examples of Memory **12** include static random access memory (SRAM), Flash memory, Burst SRAM or SynchBurst SRAM (BSRAM), Dynamic random access memory (DRAM), Fast Page Mode DRAM (FPM DRAM), Enhanced DRAM (EDRAM), Extended Data Output RAM (EDO RAM), Extended Data Output DRAM (EDO DRAM), Burst Extended Data Output DRAM (BEDO DRAM), Enhanced DRAM (EDRAM), synchronous DRAM (SDRAM), JEDEC SRAM, PC100 SDRAM, Double Data Rate SDRAM (DDR SDRAM), Enhanced SDRAM (ESDRAM), SyncLink DRAM (SLDRAM), Direct Rambus DRAM (DRDRAM), Ferroelectric RAM (FRAM), and/or others. Memory **12** can be based on any of the above described

memory chips, or any other available memory chips capable of operating as described herein. In some embodiments, processor **11** can communicate with memory **12** via a system bus **5**. In other embodiments, processor **11** can communicate directly with memory **12** via a memory port **10**.

Processor **11** can communicate directly with cache memory **14** via a connection means such as a secondary bus which may also sometimes be referred to as a backside bus. In some embodiments, processor **11** can communicate with cache memory **14** using the system bus **5**. Cache memory **14** may typically have a faster response time than main memory **12** and can include a type of memory which is considered faster than main memory **12**, such as for example SRAM, BSRAM, or EDRAM. Cache memory includes any structure such as multilevel caches, for example. In some embodiments, processor **11** can communicate with one or more I/O devices **13** via a system bus **5**. Various busses can be used to connect processor **11** to any of the I/O devices **13**, such as a VESA VL bus, an ISA bus, an EISA bus, a MicroChannel Architecture (MCA) bus, a PCI bus, a PCI-X bus, a PCI-Express bus, a NuBus, and/or others. In some embodiments, processor **11** can communicate directly with I/O device **13** via HyperTransport, Rapid I/O, or InfiniBand. In further embodiments, local busses and direct communication can be mixed. For example, processor **11** can communicate with an I/O device **13** using a local interconnect bus and communicate with another I/O device **13** directly. Similar configurations can be used for any other components described herein.

Computing Device **70** may further include alternative memory such as a SD memory slot, a USB memory stick, an optical drive such as a CD-ROM drive, a CD-R/RW drive, a DVD-ROM drive or a BlueRay disc, a hard-drive, and/or any other device comprising non-volatile memory suitable for storing data or installing application programs. Computing Device **70** may further include a storage device **27** comprising any type or form of non-volatile memory for storing an operating system (OS) such as any type or form of Windows OS, Mac OS, Unix OS, Linux OS, Android OS, iPhone OS, mobile version of Windows OS, an embedded OS, or any other OS that can operate on Computing Device **70**. Computing Device **70** may also include application programs **18**, and/or data space **19** for storing additional data or information. In some embodiments, alternative memory **16** can be used as or similar to storage device **27**. Additionally, OS **17** and/or application programs **18** can be operable from a bootable medium, such as for example, a flash drive, a micro SD card, a bootable CD or DVD, and/or other bootable medium.

Application Program **18** (also referred to as program, computer program, application, script, code, or other similar name or reference) comprises instructions that can provide functionality when executed by processor **11**. As such, Application Program **18** may be used to operate (i.e. perform operations on/with) or control a device or system. Application program **18** can be implemented in a high-level procedural or object-oriented programming language, or in a low-level machine or assembly language. Any language used can be compiled, interpreted, or otherwise translated into machine language. Application program **18** can be deployed in any form including as a stand-alone program or as a module, component, subroutine, or other unit suitable for use in a computing system. Application program **18** does not necessarily correspond to a file in a file system. A program can be stored in a portion of a file that may hold other programs or data, in a single file dedicated to the program, or in multiple files (i.e. files that store one or more modules, sub programs, or portions of code, etc.). Application Program **18** can be delivered in various forms such as, for example, executable file, library, script, plugin, addon, applet, interface, console application, web application, application service provider (ASP)-type application, operating system, and/or other forms. Application program **18** can be deployed to be executed on one computing device or on multiple computing devices (i.e. cloud, distributed, or parallel computing, etc.), or at one site or distributed across multiple sites interconnected by a communication network.

Network interface **25** can be utilized for interfacing Computing Device **70** with other devices via a network through a variety of connections including standard telephone lines, wired or wireless connections, LAN or WAN links (i.e. 802.11, T1, T3, 56 kb, X.25, etc.), broadband connections (i.e. ISDN, Frame Relay, ATM, etc.), or a combination thereof. Examples of networks include the Internet, an intranet, an extranet, a local area network (LAN), a wide area network (WAN), a personal area network (PAN), a home area network (HAN), a campus area network (CAN), a metropolitan area network (MAN), a global area network (GAN), a storage area network (SAN), virtual network, a virtual private network (VPN), Bluetooth network, a wireless network, a wireless LAN, a radio network, a HomePNA, a power line communication network, a G.hn network, an optical fiber network, an Ethernet network, an active networking network, a client-server network, a peer-to-peer network, a bus network, a star network, a ring network, a mesh network, a star-bus network, a tree network, a hierarchical topology network, and/or other networks. Network interface **25** may include a built-in network adapter, network interface card, PCMCIA network card, card bus network adapter, wireless network adapter, Bluetooth network adapter, WiFi network adapter, USB network adapter, modem, and/or any other device suitable for interfacing Computing Device **70** with any type of network capable of communication and/or operations described herein.

Still referring to FIG. **1**, I/O devices **13** may be present in various shapes or forms in Computing Device **70**. Examples of I/O device **13** capable of input include a joystick, a keyboard, a mouse, a trackpad, a trackpoint, a touchscreen, a trackball, a microphone, a drawing tablet, a glove, a tactile input device, a still or video camera, and/or other input device. Examples of I/O device **13** capable of output include a video display, a touchscreen, a projector, a glasses, a speaker, a tactile output device, and/or other output device. Examples of I/O device **13** capable of input and output include a disk drive, an optical storage device, a modem, a network card, and/or other input/output device. I/O device **13** can be interfaced with processor **11** via an I/O port **15**, for example. I/O device **13** can also be controlled by I/O control **22** in some implementations. I/O control **22** may control one or more I/O devices such as Human-machine Interface **23** (i.e. keyboard, pointing device, touchscreen, joystick, mouse, optical pen, etc.). I/O control **22** enables any type or form of a device such as, for example, a video camera or microphone to be interfaced with other components of Computing Device **70**. Furthermore, I/O device **13** may also provide storage such as or similar to storage **27**, and/or alternative memory such as or similar to alternative memory **16** in some implementations.

An output interface such as a graphical user interface, an acoustic output interface, a tactile output interface, any device driver (i.e. audio, video, or other driver), and/or other output interface or system can be utilized to process output from elements of Computing Device **70** for conveyance on

an output device such as Display **21**. In some aspects, Display **21** or other output device itself may include an output interface for processing output from elements of Computing Device **70**. Further, an input interface such as a keyboard listener, a touchscreen listener, a mouse listener, any device driver (i.e. audio, video, keyboard, mouse, touchscreen, or other driver), a speech recognizer, a video interpreter, and/or other input interface or system can be utilized to process input from Human-machine Interface **23** or other input device for use by elements of Computing Device **70**. In some aspects, Human-machine Interface **23** or other input device itself may include an input interface for processing input for use by elements of Computing Device **70**.

Computing Device **70** may include or be connected to multiple display devices **21**. Display devices **21** can each be of the same or different type or form. Computing Device **70** and/or its elements comprise any type or form of suitable hardware, programs, or a combination thereof to support, enable, or provide for the connection and use of multiple display devices **21**. In one example, Computing Device **70** includes any type or form of video adapter, video card, driver, and/or library to interface, communicate, connect, or otherwise use display devices **21**. In some aspects, a video adapter may include multiple connectors to interface to multiple display devices **21**. In other aspects, Computing Device **70** includes multiple video adapters, with each video adapter connected to one or more display devices **21**. In some embodiments, Computing Device's **70** operating system can be configured for using multiple displays **21**. In other embodiments, one or more display devices **21** can be provided by one or more other computing devices such as remote computing devices connected to Computing Device **70** via a network.

In some embodiments, I/O device **13** can be a bridge between system bus **5** and an external communication bus, such as a USB bus, an Apple Desktop Bus, an RS-232 serial connection, a SCSI bus, a FireWire bus, a FireWire 800 bus, an Ethernet bus, an AppleTalk bus, a Gigabit Ethernet bus, an Asynchronous Transfer Mode bus, a HIPPI bus, a Super HIPPI bus, a SerialPlus bus, a SCI/LAMP bus, a FibreChannel bus, a Serial Attached small computer system interface bus, and/or other bus.

Computing Device **70** can operate under the control of operating system **17**, which may support Computing Device's **70** basic functions, interface with and manage hardware resources, interface with and manage peripherals, provide common services for application programs, schedule tasks, and/or perform other functionalities. A modern operating system enables features and functionalities such as a high resolution display, graphical user interface (GUI), touchscreen, cellular network connectivity (i.e. mobile operating system, etc.), Bluetooth connectivity, WiFi connectivity, global positioning system (GPS) capabilities, mobile navigation, microphone, speaker, still picture camera, video camera, voice recorder, speech recognition, music player, video player, near field communication, personal digital assistant (PDA), and/or other features, functionalities, or applications. For example, Computing Device **70** can use any conventional operating system, any embedded operating system, any real-time operating system, any open source operating system, any video gaming operating system, any proprietary operating system, any online operating system, any operating system for mobile computing devices, or any other operating system capable of running on Computing Device **70** and performing operations described herein. Example of operating systems include Windows XP, Windows 7, Windows 8, etc. manufactured by Microsoft Corporation of Redmond, Wash.; Mac OS, iPhone OS, etc. manufactured by Apple Computer of Cupertino, Calif.; OS/2 manufactured by International Business Machines of Armonk, N.Y.; Linux, a freely-available operating system distributed by Caldera Corp. of Salt Lake City, Utah; or any type or form of a Unix operating system, among others. Any operating systems such as the ones for Android devices can similarly be utilized.

Computing Device **70** can be implemented as or be part of various different model architectures such as web services, distributed computing, grid computing, cloud computing, and/or other architectures. For example, in addition to the traditional desktop, server, or mobile operating system architectures, a cloud-based operating system can be utilized to provide the structure on which embodiments of the disclosure can be implemented. Other aspects of Computing Device **70** can also be implemented in the cloud without departing from the spirit and scope of the disclosure. For example, memory, storage, processing, and/or other elements can be hosted in the cloud. In some embodiments, Computing Device **70** can be implemented on multiple devices. For example, a portion of Computing Device **70** can be implemented on a mobile device and another portion can be implemented on wearable electronics.

Computing Device **70** can be or include any mobile device, a mobile phone, a smartphone (i.e. iPhone, Windows phone, Blackberry, Android phone, etc.), a tablet, a personal digital assistant (PDA), wearable electronics, implantable electronics, or another mobile device capable of implementing the functionalities described herein. In other embodiments, Computing Device **70** can be or include an embedded device, which can be any device or system with a dedicated function within another device or system. Embedded systems range from the simplest ones dedicated to one task with no user interface to complex ones with advanced user interface that may resemble modern desktop computer systems. Examples of devices comprising an embedded device include a mobile telephone, a personal digital assistant (PDA), a gaming device, a media player, a digital still or video camera, a pager, a television device, a set-top box, a personal navigation device, a global positioning system (GPS) receiver, a portable storage device (i.e. a USB flash drive, etc.), a digital watch, a DVD player, a printer, a microwave oven, a washing machine, a dishwasher, a gateway, a router, a hub, an automobile entertainment system, an automobile navigation system, a refrigerator, a washing machine, a factory automation device, an assembly line device, a factory floor monitoring device, a thermostat, an automobile, a factory controller, a telephone, a network bridge, and/or other devices. An embedded device can operate under the control of an operating system for embedded devices such as MicroC/OS-II, QNX, VxWorks, eCos, TinyOS, Windows Embedded, Embedded Linux, and/or other embedded device operating systems.

Various implementations of the disclosed devices, systems, and/or methods can be realized in digital electronic circuitry, integrated circuitry, logic gates, specially designed application specific integrated circuits (ASICs), field programmable gate arrays (FPGAs), computer hardware, firmware, programs, virtual machines, and/or combinations thereof including their structural, logical, and/or physical equivalents.

The disclosed devices, systems, and/or methods may include clients and servers. A client and server are generally remote from each other and typically interact via a network. The relationship of a client and server may arise by virtue of

computer programs running on their respective computers and having a client-server relationship to each other.

The disclosed devices, systems, and/or methods can be implemented in a computing system that includes a back end component, a middleware component, a front end component, or any combination thereof. The components of the system can be interconnected by any form or medium of digital data communication such as, for example, a network.

Computing Device **70** may include or be interfaced with a computer program product comprising instructions or logic encoded on a computer-readable medium. Such instructions or logic, when executed, may configure or cause a processor to perform the operations and/or functionalities disclosed herein. For example, a computer program can be provided or encoded on a computer-readable medium such as an optical medium (i.e. DVD-ROM, etc.), flash drive, hard drive, any memory, firmware, or other medium. Computer program can be installed onto a computing device to cause the computing device to perform the operations and/or functionalities disclosed herein. Machine-readable medium, computer-readable medium, or other such terms may refer to any computer program product, apparatus, and/or device for providing instructions and/or data to a programmable processor. As such, machine-readable medium includes any medium that can send or receive machine instructions as a machine-readable signal. Examples of a machine-readable medium include a volatile and/or non-volatile medium, a removable and/or non-removable medium, a communication medium, a storage medium, and/or other medium. A communication medium, for example, can transmit computer readable instructions and/or data in a modulated data signal such as a carrier wave or other transport technique, and may include any other form of information delivery medium known in art. A non-transitory machine-readable medium comprises all machine-readable media except for a transitory, propagating signal.

In some embodiments, the disclosed artificially intelligent devices, systems, and/or methods for learning and/or using visual surrounding for autonomous device operation, or elements thereof, can be implemented entirely or in part in a device (i.e. microchip, circuitry, logic gates, electronic device, computing device, special or general purpose processor, etc.) or system that comprises (i.e. hard coded, internally stored, etc.) or is provided with (i.e. externally stored, etc.) instructions for implementing VSADO functionalities. As such, the disclosed artificially intelligent devices, systems, and/or methods for learning and/or using visual surrounding for autonomous device operation, or elements thereof, may include the processing, memory, storage, and/or other features, functionalities, and embodiments of Computing Device **70** or elements thereof. Such device or system can operate on its own (i.e. standalone device or system, etc.), be embedded in another device or system (i.e. a television device, an oven, a refrigerator, a vehicle, an industrial machine, a robot, a smartphone, and/or any other device or system capable of housing the elements needed for VSADO functionalities), work in combination with other devices or systems, or be available in any other configuration. In other embodiments, the disclosed artificially intelligent devices, systems, and/or methods for learning and/or using visual surrounding for autonomous device operation, or elements thereof, may include Alternative Memory **16** that provides instructions for implementing VSADO functionalities to one or more Processors **11**. In further embodiments, the disclosed artificially intelligent devices, systems, and/or methods for learning and/or using visual surrounding for autonomous device operation, or elements thereof, can be implemented entirely or in part as a computer program and executed by one or more Processors **11**. Such program can be implemented in one or more modules or units of a single or multiple computer programs. Such program may be able to attach to or interface with, inspect, and/or take control of another application program to implement VSADO functionalities. In further embodiments, the disclosed artificially intelligent devices, systems, and/or methods for learning and/or using visual surrounding for autonomous device operation, or elements thereof, can be implemented as a network, web, distributed, cloud, or other such application accessed on one or more remote computing devices (i.e. servers, cloud, etc.) via Network Interface **25**, such remote computing devices including processing capabilities and instructions for implementing VSADO functionalities. In further embodiments, the disclosed artificially intelligent devices, systems, and/or methods for learning and/or using visual surrounding for autonomous device operation, or elements thereof, can be (1) attached to or interfaced with any computing device or application program, (2) included as a feature of an operating system, (3) built (i.e. hard coded, etc.) into any computing device or application program, and/or (4) available in any other configuration to provide its functionalities.

In yet other embodiments, the disclosed artificially intelligent devices, systems, and/or methods for learning and/or using visual surrounding for autonomous device operation, or elements thereof, can be implemented at least in part in a computer program such as Java application or program. Java provides a robust and flexible environment for application programs including flexible user interfaces, robust security, built-in network protocols, powerful application programming interfaces, database or DBMS connectivity and interfacing functionalities, file manipulation capabilities, support for networked applications, and/or other features or functionalities. Application programs based on Java can be portable across many devices, yet leverage each device's native capabilities. Java supports the feature sets of most smartphones and a broad range of connected devices while still fitting within their resource constraints. Various Java platforms include virtual machine features comprising a runtime environment for application programs. Java platforms provide a wide range of user-level functionalities that can be implemented in application programs such as displaying text and graphics, playing and recording audio content, displaying and recording visual content, communicating with another computing device, and/or other functionalities. It should be understood that the disclosed artificially intelligent devices, systems, and/or methods for learning and/or using visual surrounding for autonomous device operation, or elements thereof, are programming language, platform, and operating system independent. Examples of programming languages that can be used instead of or in addition to Java include C, C++, Cobol, Python, Java Script, Tcl, Visual Basic, Pascal, VB Script, Perl, PHP, Ruby, and/or other programming languages capable of implementing the functionalities described herein.

Where a reference to a specific file or file type is used herein, other files, file types, or formats can be substituted.

Where a reference to a data structure is used herein, it should be understood that any variety of data structures can be used such as, for example, array, list, linked list, doubly linked list, queue, tree, heap, graph, map, grid, matrix, multi-dimensional matrix, table, database, database management system (DBMS), file, neural network, and/or any other type or form of a data structure including a custom one. A

data structure may include one or more fields or data fields that are part of or associated with the data structure. A field or data field may include a data, an object, a data structure, and/or any other element or a reference/pointer thereto. A data structure can be stored in one or more memories, files, or other repositories. A data structure and/or any elements thereof, when stored in a memory, file, or other repository, may be stored in a different arrangement than the arrangement of the data structure and/or any elements thereof. For example, a sequence of elements can be stored in an arrangement other than a sequence in a memory, file, or other repository.

Where a reference to a repository is used herein, it should be understood that a repository may be or include one or more files or file systems, one or more storage locations or structures, one or more storage systems, one or more data structures or objects, one or more memory locations or structures, and/or other storage, memory, or data arrangements.

Where a reference to an interface is used herein, it should be understood that the interface comprises any hardware, device, system, program, method, and/or combination thereof that enable direct or operative coupling, connection, and/or interaction of the elements between which the interface is indicated. A line or arrow shown in the figures between any of the depicted elements comprises such interface. Examples of an interface include a direct connection, an operative connection, a wired connection (i.e. wire, cable, etc.), a wireless connection, a device, a network, a bus, a circuit, a firmware, a driver, a bridge, a program, a combination thereof, and/or others.

Where a reference to an element coupled or connected to another element is used herein, it should be understood that the element may be in communication or any other interactive relationship with the other element. Furthermore, an element coupled or connected to another element can be coupled or connected to any other element in alternate implementations. Terms coupled, connected, interfaced, or other such terms may be used interchangeably herein depending on context.

Where a reference to an element matching another element is used herein, it should be understood that the element may be equivalent or similar to the other element. Therefore, the term match or matching can refer to total equivalence or similarity depending on context.

Where a reference to a device is used herein, it should be understood that the device may include or be referred to as a system, and vice versa depending on context, since a device may include a system of elements and a system may be embodied in a device.

Where a mention of a function, method, routine, subroutine, or other such procedure is used herein, it should be understood that the function, method, routine, subroutine, or other such procedure comprises a call, reference, or pointer to the function, method, routine, subroutine, or other such procedure.

Where a mention of data, object, data structure, item, element, or thing is used herein, it should be understood that the data, object, data structure, item, element, or thing comprises a reference or pointer to the data, object, data structure, item, element, or thing.

The term collection of elements can refer to plurality of elements without implying that the collection is an element itself.

Referring to FIG. **2**, an embodiment of Device **98** comprising Unit for Learning and/or Using Visual Surrounding for Autonomous Device Operation (VSADO Unit **100**) is illustrated. Device **98** also comprises interconnected Processor **11**, Human-machine Interface **23**, Picture Capturing Apparatus **90**, Memory **12**, and Storage **27**. Processor **11** includes or executes Application Program **18**. VSADO Unit **100** comprises interconnected Artificial Intelligence Unit **110**, Acquisition Interface **120**, and Modification Interface **130**. Other additional elements can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate embodiments.

In one example, the teaching presented by the disclosure can be implemented in a device or system for learning and/or using visual surrounding for autonomous device operation. The device or system may include a processor circuit (i.e. Processor **11**, etc.) configured to execute instruction sets (i.e. Instruction Sets **526**, etc.) for operating a device. The device or system may further include a memory unit (i.e. Memory **12**, etc.) configured to store data. The device or system may further include a picture capturing apparatus (i.e. Picture Capturing Apparatus **90**, etc.) configured to capture digital pictures (i.e. Digital Pictures **525**, etc.). The device or system may further include an artificial intelligence unit (i.e. Artificial Intelligence Unit **110**, etc.). The artificial intelligence unit may be configured to receive a first digital picture from the picture capturing apparatus. The artificial intelligence unit may also be configured to receive one or more instruction sets for operating the device from the processor circuit. The artificial intelligence unit may also be configured to learn the first digital picture correlated with the one or more instruction sets for operating the device. The artificial intelligence unit may also be configured to receive a new digital picture from the picture capturing apparatus. The artificial intelligence unit may also be configured to anticipate the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture. The artificial intelligence unit may also be configured to cause the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture, the executing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the one or more instruction sets for operating the device correlated with the first digital picture, the one or more operations performed in response to the executing by the processor circuit. Any of the operations of the described elements can be performed repeatedly and/or in different orders in alternate embodiments. In some embodiments, a stream of digital pictures can be used instead of or in addition to any digital picture such as, for example, using a first stream of digital pictures instead of the first digital picture. In other embodiments, a logic circuit (i.e. Logic Circuit **250**, etc.) may be used instead of the processor circuit. In such embodiments, the one or more instruction sets for operating the device may include or be substituted with one or more inputs into or one or more outputs from the logic circuit. In further embodiments, an actuator may be included instead of or in addition to the processor circuit. In such embodiments, the one or more instruction sets for operating the device may include or be substituted with one or more inputs into the actuator. Other additional elements can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate embodiments. The device or system for learning and/or using visual surrounding for autonomous device operation may include any actions or operations of any of the disclosed methods such as methods **6100**, **6200**, **6300**, **6400**, **6500**, and/or **6600** (all later described).

Device **98** comprises any hardware, programs, or a combination thereof. Device **98** may include a system. Device **98** may include any features, functionalities, and embodiments of Computing Device **70**, or elements thereof. Examples of Device **98** include a desktop or other computer, a smartphone or other mobile computer, a vehicle, an industrial machine, a toy, a robot, a microwave or other oven, and/or any other device or machine comprising processing capabilities. Such device or machine may be built for any function or purpose examples of which are described later.

User **50** (also referred to simply as user or other similar name or reference) comprises a human user or non-human user. A non-human User **50** includes any device, system, program, and/or other mechanism for operating or controlling Device **98** and/or elements thereof. In one example, User **50** may issue an operating direction to Application Program **18** responsive to which Application Program's **18** instructions or instruction sets may be executed by Processor **11** to perform a desired operation on Device **98**. In another example, User **50** may issue an operating direction to Processor **11**, Logic Circuit **250** (later described), and/or other processing element responsive to which Processor **11**, Logic Circuit **250**, and/or other processing element may implement logic to perform a desired operation on Device **98**. User's **50** operating directions comprise any user inputted data (i.e. values, text, symbols, etc.), directions (i.e. move right, move up, move forward, copy an item, click on a link, etc.), instructions or instruction sets (i.e. manually inputted instructions or instruction sets, etc.), and/or other inputs or information. A non-human User **50** can utilize more suitable interfaces instead of, or in addition to, Human-machine Interface **23** and/or Display **21** for controlling Device **98** and/or elements thereof. Examples of such interfaces include an application programming interface (API), bridge (i.e. bridge between applications, devices, or systems, etc.), driver, socket, direct or operative connection, handle, function/routine/subroutine, and/or other interfaces.

In some embodiments, Processor **11**, Logic Circuit **250**, Application Program **18**, and/or other processing element may control or affect an actuator (not shown). Actuator comprises the functionality for implementing movements, actions, behaviors, maneuvers, and/or other mechanical or physical operations. Device **98** may include one or more actuators to enable Device **98** to perform mechanical, physical, or other operations and/or to interact with its environment. For example, an actuator can be connected to or coupled to an element such as a wheel, arm, or other element to act upon the environment. Examples of an actuator include a motor, a linear motor, a servomotor, a hydraulic element, a pneumatic element, an electro-magnetic element, a spring element, and/or other actuators. Examples of types of actuators include a rotary actuator, a linear actuator, and/or other types of actuators. In other embodiments, Processor **11**, Logic Circuit **250**, Application Program **18**, and/or other processing element may control or affect any other device or element instead of or in addition to an actuator.

Picture Capturing Apparatus **90** comprises the functionality for capturing one or more pictures, and/or other functionalities. As such, Picture Capturing Apparatus **90** can be used to capture pictures of Device's **98** surrounding. In some embodiments, Picture Capturing Apparatus **90** may be or comprises a motion picture camera that can capture streams of pictures (i.e. motion pictures, videos, etc.). In other embodiments, Picture Capturing Apparatus **90** may be or comprises a still picture camera that can capture still pictures (i.e. photographs, etc.). In further embodiments, Picture Capturing Apparatus **90** may be or comprises any other picture capturing apparatus. In general, Picture Capturing Apparatus **90** may capture any light (i.e. visible light, infrared light, ultraviolet light, x-ray light, etc.) across the electromagnetic spectrum onto a light-sensitive material. In one example, a digital Picture Capturing Apparatus **90** can utilize a charge coupled device (CCD), a CMOS sensor, and/or other electronic image sensor to capture digital pictures that can then be stored in a memory or storage, or transmitted to an element such as Artificial Intelligence Unit **110**. In another example, analog Picture Capturing Apparatus **90** can utilize an analog-to-digital converter to produce digital pictures. In some embodiments, Picture Capturing Apparatus **90** can be built, embedded, or integrated in Device **98**, VSADO Unit **100**, and/or other disclosed element. In other embodiments, Picture Capturing Apparatus **90** can be an external Picture Capturing Apparatus **90** connected with Device **98**, VSADO Unit **100**, and/or other disclosed element. In further embodiments, Picture Capturing Apparatus **90** comprises Computing Device **70** or elements thereof. In general, Picture Capturing Apparatus **90** can be implemented in any suitable configuration to provide its functionalities. Picture Capturing Apparatus **90** may capture one or more Digital Pictures **525**. Digital Picture **525** (also referred to simply as digital pictures, etc.) may include a collection of color encoded pixels or dots. Examples of file formats that can be utilized to store Digital Picture **525** include JPEG, GIF, TIFF, PNG, PDF, and/or other file formats. A stream of Digital Pictures **525** (i.e. motion picture, video, etc.) may include one or more Digital Pictures **525**. Examples of file formats that can be utilized to store a stream of Digital Pictures **525** include MPEG, AVI, FLV, MOV, RM, SWF, WMV, DivX, and/or other file formats. In some aspects, Digital Picture **525** may include or be substituted with a stream of Digital Pictures **525**, and vice versa. Therefore, the terms digital picture and stream of digital pictures may be used interchangeably herein depending on context. In some aspects, Device's **98** surrounding may include exterior of Device **98**. In other aspects, Device's **98** surrounding may include interior of Device **98** in case of hollow Device **98**, Device **98** comprising compartments or openings, and/or other variously shaped Device **98**.

VSADO Unit **100** comprises any hardware, programs, or a combination thereof. VSADO Unit **100** comprises the functionality for learning the operation of Device **98** in various visual surroundings. VSADO Unit **100** comprises the functionality for structuring and/or storing this knowledge in a knowledgebase (i.e. neural network, graph, sequences, other repository, etc.). VSADO Unit **100** comprises the functionality for enabling autonomous operation of Device **98** in various visual surroundings. VSADO Unit **100** comprises the functionality for interfacing with or attaching to Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. VSADO Unit **100** comprises the functionality for obtaining instruction sets, data, and/or other information used, implemented, and/or executed by Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. VSADO Unit **100** comprises the functionality for modifying instruction sets, data, and/or other information used, implemented, and/or executed by Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. VSADO Unit **100** comprises learning, anticipating, decision making, automation, and/or other functionalities disclosed herein. Statistical, artificial intelligence, machine learning, and/or

other models or techniques are utilized to implement the disclosed devices, systems, and methods.

When the disclosed VSADO Unit **100** functionalities are applied on Application Program **18**, Processor **11**, Logic Circuit **250** (later described), and/or other processing element of Device **98**, Device **98** may become autonomous. VSADO Unit **100** may take control from, share control with, and/or release control to Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element to implement autonomous operation of Device **98**. VSADO Unit **100** may take control from, share control with, and/or release control to Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element automatically or after prompting User **50** to allow it. In some aspects, Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element of an autonomous Device **98** may include or be provided with anticipatory instructions or instruction sets that User **50** did not issue or cause to be executed. Such anticipatory instructions or instruction sets include instruction sets that User **50** may want or is likely to issue or cause to be executed. Anticipatory instructions or instruction sets can be generated by VSADO Unit **100** or elements thereof based on the visual surrounding of Device **98**. As such, Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element of an autonomous Device **98** may include or be provided with some or all original instructions or instruction sets and/or any anticipatory instructions or instruction sets generated by VSADO Unit **100**. Therefore, autonomous Device **98** operating may include executing some or all original instructions or instruction sets and/or any anticipatory instructions or instruction sets generated by VSADO Unit **100**. In one example, VSADO Unit **100** can overwrite or rewrite the original instructions or instruction sets of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element with VSADO Unit **100**-generated instructions or instruction sets. In another example, VSADO Unit **100** can insert or embed VSADO Unit **100**-generated instructions or instruction sets among the original instructions or instruction sets of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. In a further example, VSADO Unit **100** can branch, redirect, or jump to VSADO Unit **100**-generated instructions or instruction sets from the original instructions or instruction sets of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element.

In some embodiments, autonomous Device **98** operating comprises determining, by VSADO Unit **100**, a next instruction or instruction set to be executed based on Device's **98** visual surrounding prior to the user issuing or causing to be executed the next instruction or instruction set. In yet other embodiments, autonomous application operating comprises determining, by VSADO Unit **100**, a next instruction or instruction set to be executed based on Device's **98** visual surrounding prior to the system receiving the next instruction or instruction set.

In some embodiments, autonomous Device **98** operating includes a partially or fully autonomous operating. In an example involving partially autonomous Device **98** operating, a user confirms VSADO Unit **100**-generated instructions or instruction sets prior to their execution. In an example involving fully autonomous application operating, VSADO Unit **100**-generated instructions or instruction sets are executed without user or other system confirmation (i.e. automatically, etc.).

In some embodiments, a combination of VSADO Unit **100** and other systems and/or techniques can be utilized to implement Device's **98** operation. In one example, VSADO Unit **100** may be a primary or preferred system for implementing Device's **98** operation. While operating autonomously under the control of VSADO Unit **100**, Device **98** may encounter a visual surrounding that has not been encountered or learned before. In such situations, User **50** and/or non-VSADO system may take control of Device's **98** operation. VSADO Unit **100** may take control again when Device **98** encounters a previously learned visual surrounding. Naturally, VSADO Unit **100** can learn Device's **98** operation in visual surroundings while User **50** and/or non-VSADO system is in control of Device **98**, thereby reducing or eliminating the need for future involvement of User **50** and/or non-VSADO system. In another example, User **50** and/or non-VSADO system may be a primary or preferred system for control of Device's **98** operation. While operating under the control of User **50** and/or non-VSADO system, User **50** and/or non-VSADO system may release control to VSADO Unit **100** for any reason (i.e. User **50** gets tired or distracted, non-VSADO system gets stuck or cannot make a decision, etc.), at which point Device **98** can be controlled by VSADO Unit **100**. In some designs, VSADO Unit **100** may take control in certain special visual surroundings where VSADO Unit **100** may offer superior performance even though User **50** and/or non-VSADO system may generally be preferred. Once Device **98** leaves such special visual surrounding, VSADO Unit **100** may release control to User **50** and/or a non-VSADO system. In general, VSADO Unit **100** can take control from, share control with, or release control to User **50**, non-VSADO system, and/or other system or process at any time, under any circumstances, and remain in control for any period of time as needed.

In some embodiments, VSADO Unit **100** may control one or more sub-devices, sub-systems, or elements of Device **98** while User **50** and/or non-VSADO system may control other one or more sub-devices, sub-systems, or elements of Device **98**.

It should be understood that a reference to autonomous operating of Device **98** may include autonomous operating of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element depending on context.

Acquisition Interface **120** comprises the functionality for obtaining or receiving instruction sets, data, and/or other information. Acquisition Interface **120** comprises the functionality for obtaining or receiving instruction sets, data, and/or other information from Processor **11**, Application Program **18**, Logic Circuit **250** (later described), and/or other processing element. Acquisition Interface **120** comprises the functionality for obtaining or receiving instruction sets, data, and/or other information at runtime. In some aspects, an instruction set may include any computer command, instruction, signal, or input used in Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. Therefore, the terms instruction set, command, instruction, signal, input, or other such terms may be used interchangeably herein depending on context. Acquisition Interface **120** also comprises the functionality for attaching to or interfacing with Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. In one example, Acquisition Interface **120** comprises the functionality to access and/or read runtime engine/environment, virtual machine, operating system, compiler, just-in-time (JIT) compiler, interpreter, translator, execution stack, file, object, data structure, and/or other computing system elements. In another example, Acquisition Interface **120** comprises the functionality to access and/or read

memory, storage, bus, interfaces, and/or other computing system elements. In a further example, Acquisition Interface **120** comprises the functionality to access and/or read Processor **11** registers and/or other Processor **11** elements. In a further example, Acquisition Interface **120** comprises the functionality to access and/or read inputs and/or outputs of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. In a further example, Acquisition Interface **120** comprises the functionality to access and/or read functions, methods, procedures, routines, subroutines, and/or other elements of Application Program **18**. In a further example, Acquisition Interface **120** comprises the functionality to access and/or read source code, bytecode, compiled, interpreted, or otherwise translated code, machine code, and/or other code. In a further example, Acquisition Interface **120** comprises the functionality to access and/or read values, variables, parameters, and/or other data or information. Acquisition Interface **120** also comprises the functionality for transmitting the obtained instruction sets, data, and/or other information to Artificial Intelligence Unit **110** and/or other element. As such, Acquisition Interface **120** provides input into Artificial Intelligence Unit **110** for knowledge structuring, anticipating, decision making, and/or other functionalities later in the process. Acquisition Interface **120** also comprises other disclosed functionalities.

Acquisition Interface **120** can employ various techniques for obtaining instruction sets, data, and/or other information. In one example, Acquisition Interface **120** can attach to and/or obtain Processor's **11**, Application Program's **18**, Logic Circuit's **250**, and/or other processing element's instruction sets, data, and/or other information through tracing or profiling techniques. Tracing or profiling may be used for outputting Processor's **11**, Application Program's **18**, Logic Circuit's **250**, and/or other processing element's instruction sets, data, and/or other information at runtime. For instance, tracing or profiling may include adding trace code (i.e. instrumentation, etc.) to an application and/or outputting trace information to a specific target. The outputted trace information (i.e. instruction sets, data, and/or other information, etc.) can then be provided to or recorded into a file, data structure, repository, an application, and/or other system or target that may receive such trace information. As such, Acquisition Interface **120** can utilize tracing or profiling to obtain instruction sets, data, and/or other information and provide them as input into Artificial Intelligence Unit **110**. In some aspects, instrumentation can be performed in source code, bytecode, compiled, interpreted, or otherwise translated code, machine code, and/or other code. In other aspects, instrumentation can be performed in various elements of a computing system such as memory, virtual machine, runtime engine/environment, operating system, compiler, interpreter, translator, processor registers, execution stack, program counter, and/or other elements. In yet other aspects, instrumentation can be performed in various abstraction layers of a computing system such as in software layer (i.e. Application Program **18**, etc.), in virtual machine (if VM is used), in operating system, in Processor **11**, and/or in other layers or areas that may exist in a particular computing system implementation. In yet other aspects, instrumentation can be performed at various time periods in an application's execution such as source code write time, compile time, interpretation time, translation time, linking time, loading time, runtime, and/or other time periods. In yet other aspects, instrumentation can be performed at various granularities or code segments such as some or all lines of code, some or all statements, some or all instructions or instruction sets, some or all basic blocks, some or all functions/routines/subroutines, and/or some or all other code segments.

In some embodiments, Application Program **18** can be automatically instrumented. In one example, Acquisition Interface **120** can access Application Program's **18** source code, bytecode, or machine code and select instrumentation points of interest. Selecting instrumentation points may include finding locations in the source code, bytecode, or machine code corresponding to function calls, function entries, function exits, object creations, object destructions, event handler calls, new lines (i.e. to instrument all lines of code, etc.), thread creations, throws, and/or other points of interest. Instrumentation code can then be inserted at the instrumentation points of interest to output Application Program's **18** instruction sets, data, and/or other information. In response to executing instrumentation code, Application Program's **18** instruction sets, data, and/or other information may be received by Acquisition Interface **120**. In some aspects, Application Program's **18** source code, bytecode, or machine code can be dynamically instrumented. For example, instrumentation code can be dynamically inserted into Application Program **18** at runtime.

In other embodiments, Application Program **18** can be manually instrumented. In one example, a programmer can instrument a function call by placing an instrumenting instruction immediately after the function call as in the following example.

```
Object1.moveRight(73);
traceApplication('Object1.moveRight(73);');
```

In another example, an instrumenting instruction can be placed immediately before the function call, or at the beginning, end, or anywhere within the function itself. A programmer may instrument all function calls or only function calls of interest. In a further example, a programmer can instrument all lines of code or only code lines of interest. In a further example, a programmer can instrument other elements utilized or implemented within Application Program **18** such as objects and/or any of their functions, data structures and/or any of their functions, event handlers and/or any of their functions, threads and/or any of their functions, and/or other elements or functions. Similar instrumentation as in the preceding examples can be performed automatically or dynamically. In some designs where manual code instrumentation is utilized, Acquisition Interface **120** can optionally be omitted and Application Program's **18** instruction sets, data, and/or other information may be transmitted directly to Artificial Intelligence Unit **110**.

In some embodiments, VSADO Unit **100** can be selective in learning instruction sets, data, and/or other information to those implemented, utilized, or related to an object, data structure, repository, thread, function, and/or other element of Application Program **18**. In some aspects, Acquisition Interface **120** can obtain Application Program's **18** instruction sets, data, and/or other information implemented, utilized, or related to a certain object in an object oriented Application Program **18**.

In some embodiments, various computing systems and/or platforms may provide native tools for obtaining instruction sets, data, and/or other information. Also, independent vendors may provide portable tools with similar functionalities that can be utilized across different computing systems and/or platforms. These native and portable tools may provide a wide range of functionalities to obtain runtime and other information such as instrumentation, tracing or profiling, logging application or system messages, outputting

custom text messages, outputting objects or data structures, outputting functions/routines/subroutines or their invocations, outputting variable or parameter values, outputting thread or process behaviors, outputting call or other stacks, outputting processor registers, providing runtime memory access, providing inputs and/or outputs, performing live application monitoring, and/or other capabilities. One of ordinary skill in art will understand that, while all possible variations of the techniques to obtain instruction sets, data, and/or other information are too voluminous to describe, these techniques are within the scope of this disclosure.

In one example, obtaining instruction sets, data, and/or other information can be implemented through the .NET platform's native tools for application tracing or profiling such as System.Diagnostics.Trace, System.Diagnostics.Debug, and System.Diagnostics.TraceSource classes for tracing execution flow, and System. Diagnostics. Process, System.Diagnostics.EventLog, and System. Diagnostics. PerformanceCounter classes for profiling code, accessing local and remote processes, starting and stopping system processes, and interacting with Windows event logs, etc. For instance, a set of trace switches can be created that output an application's information. The switches can be configured using the .config file. For a Web application, this may typically be Web.config file associated with the project. In a Windows application, this file may typically be named applicationName.exe.config. Trace code can be added to application code automatically or manually as previously described. Appropriate listener can be created where the trace output is received. Trace code may output trace messages to a specific target such as a file, a log, a database, an object, a data structure, and/or other repository or system. Acquisition Interface **120** or Artificial Intelligence Unit **110** can then read or obtain the trace information from these targets. In some aspects, trace code may output trace messages directly to Acquisition Interface **120**. In other aspects, trace code may output trace messages directly to Artificial Intelligence Unit **110**. In the case of outputting trace messages to Acquisition Interface **120** or directly to Artificial Intelligence Unit **110**, custom listeners can be built to accommodate these specific targets. Other platforms, tools, and/or techniques can provide equivalent or similar functionalities as the above described ones.

In another example, obtaining instruction sets, data, and/or other information can be implemented through the .NET platform's Profiling API that can be used to create a custom profiler application for tracing, monitoring, interfacing with, and/or managing a profiled application. The Profiling API provides an interface that includes methods to notify the profiler of events in the profiled application. The Profiling API may also provide an interface to enable the profiler to call back into the profiled application to obtain information about the state of the profiled application. The Profiling API may further provide call stack profiling functionalities. Call stack (also referred to as execution stack, control stack, runtime stack, machine stack, the stack, etc.) includes a data structure that can store information about active subroutines of an application. The Profiling API may provide a stack snapshot method, which enables a trace of the stack at a particular point in time. The Profiling API may also provide a shadow stack method, which tracks the call stack at every instant. A shadow stack can obtain function arguments, return values, and information about generic instantiations. A function such as FunctionEnter can be utilized to notify the profiler that control is being passed to a function and can provide information about the stack frame and function arguments. A function such as FunctionLeave can be utilized to notify the profiler that a function is about to return to the caller and can provide information about the stack frame and function return value. An alternative to call stack profiling includes call stack sampling in which the profiler can periodically examine the stack. In some aspects, the Profiling API enables the profiler to change the in-memory code stream for a routine before it is just-in-time (JIT) compiled where the profiler can dynamically add instrumentation code to all or particular routines of interest. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through Java platform's APIs for application tracing or profiling such as Java Virtual Machine Profiling Interface (JVMPI), Java Virtual Machine Tool Interface (JVMTI), and/or other APIs or tools. These APIs can be used for instrumentation of an application, for notification of Java Virtual Machine (VM) events, and/or other functionalities. One of the tracing or profiling techniques that can be utilized includes bytecode instrumentation. The profiler can insert bytecodes into all or some of the classes. In application execution profiling, for example, these bytecodes may include methodEntry and methodExit calls. In memory profiling, for example, the bytecodes may be inserted after each new or after each constructor. In some aspects, insertion of instrumentation bytecode can be performed either by a post-compiler or a custom class loader. An alternative to bytecode instrumentation includes monitoring events generated by the JVMPI or JVMTI interfaces. Both APIs can generate events for method entry/exit, object allocation, and/or other events. In some aspects, JVMTI can be utilized for dynamic bytecode instrumentation where insertion of instrumentation bytecodes is performed at runtime. The profiler may insert the necessary instrumentation when a selected class is invoked in an application. This can be accomplished using the JVMTI's redefineClasses method, for example. This approach also enables changing of the level of profiling as the application is running. If needed, these changes can be made adaptively without restarting the application. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through JVMTI's programming interface that enables creation of software agents that can monitor and control a Java application. An agent may use the functionality of the interface to register for notification of events as they occur in the application, and to query and control the application. A JVMTI agent may use JVMTI functions to extract information from a Java application. A JVMTI agent can be utilized to obtain an application's runtime information such as method calls, memory allocation, CPU utilization, lock contention, and/or other information. JVMTI may include functions to obtain information about variables, fields, methods, classes, and/or other information. JVMTI may also provide notification for numerous events such as method entry and exit, exception, field access and modification, thread start and end, and/or other events. Examples of JVMTI built-in methods include GetMethodName to obtain the name of an invoked method, GetThreadInfo to obtain information for a specific thread, GetClassSignature to obtain information about the class of an object, GetStackTrace to obtain information about the stack including information about stack frames, and/or other methods. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through java.lang.Runtime class that provides an interface for application tracing or profiling. Examples of methods provided in java.lang.Runtime that can be used to obtain an application's instruction sets, data, and/or other information include tracemethodcalls, traceinstructions, and/or other methods. These methods prompt the Java Virtual Machine to output trace information for a method or instruction in the virtual machine as it is executed. The destination of trace output may be system dependent and include a file, a listener, and/or other destinations where Acquisition Interface **120**, Artificial Intelligence Unit **110**, and/or other disclosed elements can access needed information. In addition to tracing or profiling tools native to their respective computing systems and/or platforms, many independent tools exist that provide tracing or profiling functionalities on more than one computing system and/or platform. Examples of these tools include Pin, DynamoRIO, KernInst, DynInst, Kprobes, OpenPAT, DTrace, SystemTap, and/or others. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through logging tools of the platform and/or operating system on which an application runs. Some logging tools may include nearly full feature sets of the tracing or profiling tools previously described. In one example, Visual Basic enables logging of runtime messages through its Microsoft.VisualBasic.Logging namespace that provides a log listener where the log listener may direct logging output to a file and/or other target. In another example, Java enables logging through its java.util.logging class. In some aspects, obtaining an application's instruction sets, data, and/or other information can be implemented through logging capabilities of the operating system on which an application runs. For example, Windows NT features centralized log service that applications and operating-system components can utilize to report their events including any messages. Windows NT provides functionalities for system, application, security, and/or other logging. An application log may include events logged by applications. Windows NT, for example, may include support for defining an event source (i.e. application that created the event, etc.). Windows Vista, for example, supports a structured XML log-format and designated log types to allow applications to more precisely log events and to help interpret the events. Examples of different types of event logs include administrative, operational, analytic, debug, and/or other log types including any of their subcategories. Examples of event attributes that can be utilized include eventID, level, task, opcode, keywords, and/or other event attributes. Windows wevtutil tool enables access to events, their structures, registered event publishers, and/or their configuration even before the events are fired. Wevtutil supports capabilities such as retrieval of the names of all logs on a computing device; retrieval of configuration information for a specific log; retrieval of event publishers on a computing device; reading events from an event log, from a log file, or using a structured query; exporting events from an event log, from a log file, or using a structured query to a specific target; and/or other capabilities. Operating system logs can be utilized solely if they contain sufficient information on an application's instruction sets, data, and/or other information. Alternatively, operating system logs can be utilized in combination with another source of information (i.e. trace information, call stack, processor registers, memory, etc.) to reconstruct the application's instruction sets, data, and/or other information needed for Artificial Intelligence Unit **110** and/or other elements. In addition to logging capabilities native to their respective platforms and/or operating systems, many independent tools exist that provide logging on different platforms and/or operating systems. Examples of these tools include Log 4j, Logback, SmartInspect, NLog, log 4net, Microsoft Enterprise Library, ObjectGuy Framework, and/or others. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through tracing or profiling the operating system on which an application runs. As in tracing or profiling an application, one of the techniques that can be utilized includes adding instrumentation code to the operating system's source code. Such instrumentation code can be added to the operating system's source code before kernel compilation or recompilation, for instance. This type of instrumentation may involve defining or finding locations in the operating system's source code where instrumentation code may be inserted. Kernel instrumentation can also be performed without the need for kernel recompilation or rebooting. In some aspects, instrumentation code can be added at locations of interest through binary rewriting of compiled kernel code. In other aspects, kernel instrumentation can be performed dynamically where instrumentation code is added and/or removed where needed at runtime. For instance, dynamic instrumentation may overwrite kernel code with a branch instruction that redirects execution to instrumentation code or instrumentation routine. In yet other aspects, kernel instrumentation can be performed using just-in-time (JIT) dynamic instrumentation where execution may be redirected to a copy of kernel's code segment that includes instrumentation code. This type of instrumentation may include a JIT compiler and creation of a copy of the original code segment having instrumentation code or calls to instrumentation routines embedded into the original code segment. Instrumentation of the operating system may enable total system visibility including visibility into an application's behavior by enabling generation of low level trace information. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through tracing or profiling the processor on which an application runs. For example, some Intel processors provide Intel Processor Trace (i.e. Intel PT, etc.), a low-level tracing feature that enables recording executed instruction sets, and/or other data or information of one or more applications. Intel PT is facilitated by the Processor Trace Decoder Library along with its related tools. Intel PT is a low-overhead execution tracing feature that records information about application execution on each hardware thread using dedicated hardware facilities. The recorded execution/trace information is collected in data packets that can be buffered internally before being sent to a memory subsystem or another system or element (i.e. Acquisition Interface **120**, Artificial Intelligence Unit **110**, etc.). Intel PT also enables navigating the recorded execution/trace information via reverse stepping commands. Intel PT can be included in an operating system's core files and provided as a feature of the operating system. Intel PT can trace globally some or all applications running on an operating system. Acquisition Interface **120** or Artificial Intelligence Unit **110** can read or obtain the recorded execution/trace information from Intel PT. Other

platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through branch tracing or profiling. Branch tracing may include an abbreviated instruction trace in which only the successful branch instruction sets are traced or recorded. Branch tracing can be implemented through utilizing dedicated processor commands, for example. Executed branches may be saved into special branch trace store area of memory. With the availability and reference to a compiler listing of the application together with branch trace information, a full path of executed instruction sets can be reconstructed. The full path can also be reconstructed with a memory dump (containing the program storage) and branch trace information. In some aspects, branch tracing can be utilized for pre-learning or automated learning of an application's instruction sets, data, and/or other information where a number of application simulations (i.e. simulations of likely/common operations, etc.) are performed. As such, the application's operation can be learned automatically saving the time that would be needed to learn the application's operation directed by a user. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through assembly language. Assembly language is a low-level programming language for a computer or other programmable device in which there is a strong correlation between the language and the architecture's machine instruction sets. Syntax, addressing modes, operands, and/or other elements of an assembly language instruction set may translate directly into numeric (i.e. binary, etc.) representations of that particular instruction set. Because of this direct relationship with the architecture's machine instruction sets, assembly language can be a powerful tool for tracing or profiling an application's execution in processor registers, memory, and/or other computing system components. For example, using assembly language, memory locations of a loaded application can be accessed, instrumented, and/or otherwise manipulated. In some aspects, assembly language can be used to rewrite or overwrite original in-memory instruction sets of an application with instrumentation instruction sets. In other aspects, assembly language can be used to redirect application's execution to instrumentation routine/subroutine or other code segment elsewhere in memory by inserting a jump into the application's in-memory code, by redirecting program counter, or by other techniques. Some operating systems may implement protection from changes to applications loaded into memory. Operating system, processor, or other low level commands such as Linux mprotect command or similar commands in other operating systems may be used to unprotect the protected locations in memory before the change. In yet other aspects, assembly language can be used to obtain instruction sets, data, and/or other information through accessing and/or reading instruction register, program counter, other processor registers, memory locations, and/or other components of a computing system. In yet other aspects, high-level programming languages may call or execute an external assembly language program to facilitate obtaining instruction sets, data, and/or other information as previously described. In yet other aspects, relatively low-level programming languages such as C may allow embedding assembly language directly in their source code such as, for example, using asm keyword of C. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, it may be sufficient to obtain user or other inputs, variables, parameters, and/or other data in some procedural, simple object oriented, or other applications. In one instance, a simple procedural application executes a sequence of instruction sets until the end of the program. During its execution, the application may receive user or other input, store the input in a variable, and perform calculations using the variable to reach a result. The value of the variable can be obtained or traced. In another instance, a more complex procedural application comprises one or more functions/routines/subroutines each of which may include a sequence of instruction sets. The application may execute a main sequence of instruction sets with a branch to a function/routine/subroutine. During its execution, the application may receive user or other input, store the input in a variable, and pass the variable as a parameter to the function/routine/subroutine. The function/routine/subroutine may perform calculations using the parameter and return a value that the rest of the application can use to reach a result. The value of the variable or parameter passed to the function/routine/subroutine, and/or return value can be obtained or traced. Values of user or other inputs, variables, parameters, and/or other items of interest can be obtained through previously described tracing, instrumentation, and/or other techniques. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

Referring to FIG. **3**, in yet another example, obtaining instruction sets, data, and/or other information may be implemented through tracing, profiling, or sampling of instruction sets or data in processor registers, memory, or other computing system components where instruction sets, data, and/or other information may be stored or utilized. For example, Instruction Register **212** may be part of Processor **11** and it may store the instruction set currently being executed or decoded. In some processors, Program Counter **211** (also referred to as instruction pointer, instruction address register, instruction counter, or part of instruction sequencer) may be incremented after fetching an instruction set, and it may hold or point to the memory address of the next instruction set to be executed. In a processor where the incrementation precedes the fetch, Program Counter **211** may point to the current instruction set being executed. In the instruction cycle, an instruction set may be loaded into Instruction Register **212** after Processor **11** fetches it from location in Memory **12** pointed to by Program Counter **211**. Instruction Register **212** may hold the instruction set while it is decoded by Instruction Decoder **213**, prepared, and executed. In some aspects, data (i.e. operands, etc.) needed for instruction set execution may be loaded from Memory **12** into a register within Register Array **214**. In other aspects, the data may be loaded directly into Arithmetic Logic Unit **215**. For instance, as instruction sets pass through Instruction Register **212** during application execution, they may be transmitted to Acquisition Interface **120** as shown. Examples of the steps in execution of a machine instruction set may include decoding the opcode (i.e. portion of a machine instruction set that may specify the operation to be performed), determining where the operands may be located (depending on architecture, operands may be in registers, the stack, memory, I/O ports, etc.), retrieving the operands, allocating processor resources to execute the instruction set (needed in some types of processors), performing the operation indicated by the instruction set, saving the results of execution, and/or other execution steps. Examples of the

types of machine instruction sets that can be utilized include arithmetic, data handling, logical, program control, as well as special and/or other instruction set types. In addition to the ones described or shown, examples of other computing system or processor components that can be used during an instruction cycle include memory address register (MAR) that may hold the address of a memory block to be read from or written to; memory data register (MDR) that may hold data fetched from memory or data waiting to be stored in memory; data registers that may hold numeric values, characters, small bit arrays, or other data; address registers that may hold addresses used by instruction sets that indirectly access memory; general purpose registers (GPRs) that may store both data and addresses; conditional registers that may hold truth values often used to determine whether some instruction set should or should not be executed; floating point registers (FPRs) that may store floating point numbers; constant registers that may hold read-only values such as zero, one, or pi; special purpose registers (SPRs) such as status register, program counter, or stack pointer that may hold information on program state; machine-specific registers that may store data and settings related to a particular processor; Register Array **214** that may include an array of any number of processor registers; Arithmetic Logic Unit **215** that may perform arithmetic and logic operations; control unit that may direct processor's operation; and/or other circuits or components. Tracing, profiling, or sampling of processor registers, memory, or other computing system components can be implemented in a program, combination of hardware and program, or purely hardware system. Dedicated hardware may be built to perform tracing, profiling, or sampling of processor registers or any computing system components with marginal or no impact to computing overhead.

One of ordinary skill in art will recognize that FIG. **3** depicts one of many implementations of processor or computing system components, and that various additional components can be included, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate implementations. Processor or computing system components may be arranged or connected differently in alternate implementations. Processor or computing system components may also be connected with external elements using various connections. For instance, the connection between Instruction Register **212** and Acquisition Interface **120** may include any number or types of connections such as, for example, a dedicated connection for each bit of Instruction Register **212** (i.e. 32 connections for a 32 bit Instruction Register **212**, etc.). Any of the described or other connections or interfaces may be implemented among any processor or computing system components and Acquisition Interface **120** or other elements.

Referring to FIGS. **4**A-**4**B, in yet another example, obtaining instruction sets, data, and/or other information may be implemented through tracing, profiling, or sampling of Logic Circuit **250**. While Processor **11** includes any type or embodiment of logic circuit, Logic Circuit **250** is described separately here to offer additional detail on its functioning. Some Devices **98** may not need the processing capabilities of an entire Processor **11**, but instead a more tailored Logic Circuit **250**. Examples of such Devices **98** include home appliances, audio or video electronics, vehicle systems, toys, industrial machines, robots, and/or others. Logic Circuit **250** comprises the functionality for performing logic operations. Logic Circuit **250** comprises the functionality for performing logic operations using the circuit's inputs and producing outputs based on the logic operations performed on the inputs. Logic Circuit **250** may generally be implemented using transistors, diodes, and/or other electronic switches, but can also be constructed using vacuum tubes, electromagnetic relays (relay logic), fluidic logic, pneumatic logic, optics, molecules, or even mechanical elements. In some aspects, Logic Circuit **250** may be or include a microcontroller, field-programmable gate array (FPGA), application-specific integrated circuit (ASIC), and/or other computing circuit or device. In other aspects, Logic Circuit **250** may be or include any circuit or device comprising one or more logic gates, one or more transistors, one or more switches, and/or one or more other logic components. In further aspects, Logic Circuit **250** may be or include any integrated or other circuit or device that can perform logic operations. Logic may generally refer to Boolean logic utilized in binary operations, but other logics can also be used. Input into Logic Circuit **250** may include or refer to a value inputted into the Logic Circuit **250**, therefore, these terms may be used interchangeably herein depending on context. In one example, Logic Circuit **250** may perform some logic operations using four input values and produce two output values. As the four input values are delivered to or received by Logic Circuit **250**, they may be obtained by Acquisition Interface **120** through the four hardwired connections as shown in FIG. **4**A. In another example, Logic Circuit **250** may perform some logic operations using four input values and produce two output values. As the two output values are generated by or transmitted out of Logic Circuit **250**, they may be obtained by Acquisition Interface **120** through the two hardwired connections as shown in FIG. **4**B. In a further example, instead of or in addition to obtaining input and/or output values of Logic Circuit **250**, the state of Logic Circuit **250** may be obtained by reading or accessing values from one or more Logic Circuit's **250** internal components such as registers, memories, buses, and/or others (i.e. similar to the previously described tracing, profiling, and/or sampling of Processor **11** components, etc.). Tracing, profiling, or sampling of Logic Circuit **250** can be implemented in a program, combination of hardware and program, or purely hardware system. Dedicated hardware may be built to perform tracing, profiling or sampling of Logic Circuit **250** with marginal or no impact to computing overhead. Any of the elements and/or techniques for tracing, profiling, or sampling of Logic Circuit **250** can similarly be implemented with Processor **11** and/or other processing elements. In some designs, VSADO Unit **100** may include clamps and/or other elements to attach VSADO Unit **100** to inputs (i.e. input wires, etc.) into and/or outputs (i.e. output wires, etc.) from Logic Circuit **250**. Such clamps and/or attachment elements enable seamless attachment of VSADO Unit **100** to any circuit or computing device without the need to redesign or alter the circuit or computing device.

In some embodiments, VSADO Unit **100** may learn input values directly from an actuator (previously described, not shown). For example, Logic Circuit **250** or other processing element may control an actuator that enables Device **98** to perform mechanical, physical, and/or other operations. An actuator may receive one or more input values or control signals from Logic Circuit **250** or other processing element directing the actuator to perform specific operations. As one or more input values or control signals are delivered to or received by the actuator, they may be obtained by Acquisition Interface **120** as previously described with respect to obtaining input values of Logic Circuit **250**. Specifically, for instance, one or more input values or control signals of an

actuator may be obtained by Acquisition Interface **120** via hardwired or other connections.

One of ordinary skill in art will recognize that FIGS. **4**A-**4**B depict one of many implementations of Logic Circuit **250** and that any number of input and/or output values can be utilized in alternate implementations. One of ordinary skill in art will also recognize that Logic Circuit **250** may include any number and/or combination of logic components to implement any logic operations.

Other additional techniques or elements may be utilized as needed for obtaining instruction sets, data, and/or other information, or some of the disclosed techniques or elements may be excluded, or a combination thereof may be utilized in alternate embodiments.

Referring to FIGS. **5**A-**5**C, some embodiments of Instruction Sets **526** are illustrated. In some aspects, Instruction Set **526** includes one or more instructions or commands of Application Program **18**. For example, Instruction Set **526** may include one or more instructions or commands of a high-level programming language such as Java or SQL, a low-level language such as assembly or machine language, an intermediate language or construct such as bytecode, and/or any other language or construct. In other aspects, Instruction Set **526** includes one or more inputs into and/or outputs from Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element. In further aspects, Instruction Set **526** includes one or more values or states of registers and/or other components of Logic Circuit **250**, Processor **11**, and/or other processing element. In general, Instruction Set **526** may include one or more instructions, commands, keywords, symbols (i.e. parentheses, brackets, commas, semicolons, etc.), operators (i.e. =, <, >, etc.), variables, values, objects, data structures, functions (i.e. Function1( ), FIRST( ), MIN( ), SQRT( ), etc.), parameters, states, signals, inputs, outputs, characters, digits, references thereto, and/or other components for performing an operation.

In an embodiment shown in FIG. **5**A, Instruction Set **526** includes code of a high-level programming language (i.e. Java, C++, etc.) comprising the following function call construct: Function1 (Parameter1, Parameter2, Parameter3, . . .). An example of a function call applying the above construct includes the following Instruction Set **526**: moveTo(Object1, 29, 17). The function or reference thereto "moveTo(Object1, 29, 17)" may be an Instruction Set **526** directing Object1 to move to a location with coordinates 29 and 17, for example. In another embodiment shown in FIG. **5**B, Instruction Set **526** includes structured query language (SQL). In a further embodiment shown in FIG. **5**C, Instruction Set **526** includes bytecode (i.e. Java bytecode, Python bytecode, CLR bytecode, etc.). In a further embodiment shown in FIG. **5**D, Instruction Set **526** includes assembly code. In a further embodiment shown in FIG. **5**E, Instruction Set **526** includes machine code.

Referring to FIGS. **6**A-**6**B, some embodiments of Extra Information **527** (also referred to as Extra Info **527**) are illustrated. In an embodiment shown in FIG. **6**A, Digital Picture **525** may include or be associated with Extra Info **527**. In an embodiment shown in FIG. **6**B, Instruction Set **526** may include or be associated with Extra Info **527**.

Extra Info **527** comprises the functionality for storing any information useful in comparisons or decision making performed in autonomous device operation, and/or other functionalities. One or more Extra Infos **527** can be stored in, appended to, or associated with a Digital Picture **525**, Instruction Set **526**, and/or other element. In some embodiments, the system can obtain Extra Info **527** at a time of

capturing or receiving of Digital Picture **525**. In other embodiments, the system can obtain Extra Info **527** at a time of acquisition of Instruction Set **526**. In general, the system or any element thereof can obtain Extra Info **527** at any time. Examples of Extra Info **527** include time information, location information, computed information, observed information, sensory information, contextual information, and/or other information. Any information can be utilized that can provide information for enhanced comparisons or decision making performed in autonomous device operation. Which information is utilized and/or stored in Extra Info **527** can be set by a user, by VSADO system administrator, or automatically by the system. Extra Info **527** may include or be referred to as contextual information, and vice versa. Therefore, these terms may be used interchangeably herein depending on context.

In some aspects, time information (i.e. time stamp, etc.) can be utilized and/or stored in Extra Info **527**. Time information can be useful in comparisons or decision making performed in autonomous device operation related to a specific time period as Device **98** may be required to perform specific operations at certain parts of day, month, year, and/or other time periods. Time information can be obtained from the system clock, online clock, oscillator, or other time source. In one example, a thermostat device may be directed to turn heat on in the morning and/or turn heat off during the day. In a further example, a personal computer device may be directed to start or stop an application program or process on a particular day of the month. In general, Extra Info **527** may include time information related to when Device **98** performed an operation. In other aspects, location information (i.e. coordinates, address, etc.) can be utilized and/or stored in Extra Info **527**. Location information can be useful in comparisons or decision making performed in autonomous device operation related to a specific place as Device **98** may be required to perform specific operations at certain places. Location information can be obtained from a positioning system (i.e. radio signal triangulation in smartphones or tablets, GPS capabilities, etc.) if one is available. In one example, a smartphone device may be directed to engage a vibrate mode in a school or house of worship. In another example, a vehicle may be directed to turn right at a particular road crossing. In general, Extra Info **527** may include location information related to where Device **98** performed an operation. In further aspects, computed information can be utilized and/or stored in Extra Info **527**. Computed information can be useful in comparisons or decision making performed in autonomous device operation where information can be calculated, inferred, or derived from other available information. VSADO Unit **100** may include computational functionalities to create Extra Info **527** by performing calculations or inferences using other information. In one example, Device's **98** speed can be computed or estimated from the Device's **98** location and/or time information. In another example, Device's **98** bearing (i.e. angle or direction of movement, etc.) can be computed or estimated from the Device's **98** location information by utilizing Pythagorean theorem, trigonometry, and/or other theorems, formulas, or disciplines. In a further example, speeds, bearings, distances, and/or other properties of objects around Device **98** can similarly be computed or inferred, thereby providing geo-spatial and situational awareness and/or capabilities to the Device **98**. In further aspects, observed information can be utilized and/or stored in Extra Info **527**. Observed information can be useful in comparisons or decision making performed in autonomous device operation related to a specific object or environment

as Device **98** may be required to perform certain operations around specific objects or in specific environments. For example, an object or environment can be recognized by processing one or more Digital Pictures **525** from Picture Capturing Apparatus **90**. Any features, functionalities, and embodiments of Picture Recognizer **350** (later described) can be utilized for such recognizing. In one example, book shelves recognized in the background of one or more Digital Pictures **525** from Picture Capturing Apparatus **90** may indicate a library or book store. In another example, trees recognized in the background of one or more Digital Pictures **525** from Picture Capturing Apparatus **90** may indicate a park. In a further example, a pedestrian recognized in one or more Digital Pictures **525** from Picture Capturing Apparatus **90** may indicate a street. In further aspects, sensory information can be utilized and/or stored in Extra Info **527**. Examples of sensory information include acoustic information, visual information, tactile information, and/or others. Sensory information can be useful in comparisons or decision making performed in autonomous device operation related to a specific object or environment as Device **98** may be required to perform certain operations around specific objects or in specific environments. For example, an object or environment can be recognized by processing digital sound from a sound capturing apparatus (i.e. microphone, etc., not shown). Any features, functionalities, and embodiments of a speech or sound recognizer (not shown) can be utilized for such recognizing. In one example, sound of waves recognized in digital sound from a sound capturing apparatus may indicate a beach. In another example, sound of a horn recognized in digital sound from a sound capturing apparatus may indicate a proximal vehicle. In some designs where acoustic information includes one or more digital sound samples of Device's **98** surrounding captured by a sound capturing apparatus, the digital sound samples can be learned and/or used similar to Digital Pictures **525** of Device's **98** visual surrounding. In such designs, both Digital Pictures **525** and digital sound samples of a device's surrounding can be learned and/or used for autonomous device operation. In further aspects, other information can be utilized and/or stored in Extra Info **527**. Examples of such other information include user specific information (i.e. skill level, age, gender, etc.), group user information (i.e. access level, etc.), version of Application Program **18**, the type of Application Program **18**, the type of Processor **11**, the type of Logic Circuit **250**, the type of Device **98**, and/or other information.

Referring to FIG. **7**, an embodiment where VSADO Unit **100** is part of or operating on Processor **11** is illustrated. In one example, VSADO Unit **100** may be a hardware element or circuit embedded or built into Processor **11**. In another example, VSADO Unit **100** may be a program operating on Processor **11**.

Referring to FIG. **8**, an embodiment where VSADO Unit **100** resides on Server **96** accessible over Network **95** is illustrated. Any number of Devices **98** may connect to such remote VSADO Unit **100** and the remote VSADO Unit **100** may learn their operations in various visual surroundings. In turn, any number of Devices **98** can utilize the remote VSADO Unit **100** for autonomous operation. A remote VSADO Unit **100** can be offered as a network service (i.e. online application, etc.). In some aspects, a remote VSADO Unit **100** (i.e. global VSADO Unit **100**, etc.) may reside on the Internet and be available to all the world's Devices **98** configured to transmit their operations in various visual surroundings and/or configured to utilize the remote VSADO Unit **100** for autonomous operation. Server **96** may be or include any type or form of a remote computing device such as an application server, a network service server, a cloud server, a cloud, and/or other remote computing device. Server **96** may include any features, functionalities, and embodiments of the previously described Computing Device **70**. It should be understood that Server **96** does not have to be a separate computing device and that Server **96**, its elements, or its functionalities can be implemented on Device **98**. Network **95** may include various networks, connection types, protocols, interfaces, APIs, and/or other elements or techniques known in art all of which are within the scope of this disclosure. Any of the previously described networks, network or connection types, networking interfaces, and/or other networking elements or techniques can similarly be utilized. Any of the disclosed elements may reside on Server **96** in alternate implementations. In one example, Artificial Intelligence Unit **110** can reside on Server **96** and Acquisition Interface **120** and/or Modification Interface **130** can reside on Device **98**. In another example, Knowledgebase **530** can reside on Server **96** and the rest of the elements of VSADO Unit **100** can reside on Device **98**. Any other combination of local and remote elements can be implemented.

Referring to FIG. **9**, an embodiment where Picture Capturing Apparatus **90** is part of Remote Device **97** accessible over Network **95** is illustrated. In such embodiments, VSADO Unit **100** may learn Device's **98** operation based on another device's visual surrounding. Such embodiments can be utilized, for instance, in any situation where one device controls (i.e. remote control, etc.) another device, any situation where some or all of the processing is on one device and picture capturing capabilities are on another device, and/or other situations. In one example, a drone controlling device (i.e. Device **98**) may receive its visual input from a camera on the drone (i.e. Remote Device **97**). In another example, a toy controlling device (i.e. Device **98**) may receive its visual input from a camera on the toy (i.e. Remote Device **97**). In a further example, a people or crowd analyzing computing device (i.e. Device **98**) may receive its visual input from a camera of a monitoring device (i.e. Remote Device **97**). Any of the disclosed elements in addition to Picture Capturing Apparatus **90** may reside on Remote Device **97** in alternate implementations as previously described with respect to Server **96**.

Referring to FIG. **10**, an embodiment of VSADO Unit **100** comprising Picture Recognizer **350** is illustrated. VSADO Unit **100** can utilize Picture Recognizer **350** to detect or recognize persons, objects, and/or their activities in one or more digital pictures from Picture Capturing Apparatus **90**. In general, VSADO Unit **100** and/or other disclosed elements can use Picture Recognizer **350** for any operation supported by Picture Recognizer **350**. Picture Recognizer **350** comprises the functionality for detecting or recognizing persons or objects in visual data. Picture Recognizer **350** comprises the functionality for detecting or recognizing activities in visual data. Picture Recognizer **350** comprises the functionality for tracking persons, objects, and/or their activities in visual data. Picture Recognizer **350** comprises other disclosed functionalities. Visual data includes digital motion pictures, digital still pictures (i.e. bitmaps, etc.), and/or other visual data. Examples of file formats that can be utilized to store visual data include AVI, DivX, MPEG, JPEG, GIF, TIFF, PNG, PDF, and/or other file formats. Picture Recognizer **350** may detect or recognize a person and/or his/her activities as well as track the person and/or his/her activities in one or more digital pictures or streams of digital pictures (i.e. motion pictures, video, etc.). Picture

Recognizer **350** may detect or recognize a human head or face, upper body, full body, or portions/combinations thereof. In some aspects, Picture Recognizer **350** may detect or recognize persons, objects, and/or their activities from a digital picture by comparing regions of pixels from the digital picture with collections of pixels comprising known persons, objects, and/or their activities. The collections of pixels comprising known persons, objects, and/or their activities can be learned or manually, programmatically, or otherwise defined. The collections of pixels comprising known persons, objects, and/or their activities can be stored in any data structure or repository (i.e. one or more files, database, etc.) that resides locally on Device **98**, or remotely on a remote computing device (i.e. server, cloud, etc.) accessible over a network. In other aspects, Picture Recognizer **350** may detect or recognize persons, objects, and/or their activities from a digital picture by comparing features (i.e. lines, edges, ridges, corners, blobs, regions, etc.) of the digital picture with features of known persons, objects, and/or their activities. The features of known persons, objects, and/or their activities can be learned or manually, programmatically, or otherwise defined. The features of known persons, objects, and/or their activities can be stored in any data structure or repository (i.e. neural network, one or more files, database, etc.) that resides locally on Device **98**, or remotely on a remote computing device (i.e. server, cloud, etc.) accessible over a network. Typical steps or elements in a feature oriented picture recognition include pre-processing, feature extraction, detection/segmentation, decision-making, and/or others, or a combination thereof, each of which may include its own sub-steps or sub-elements depending on the application. In further aspects, Picture Recognizer **350** may detect or recognize multiple persons, objects, and/or their activities from a digital picture using the aforementioned pixel or feature comparisons, and/or other detection or recognition techniques. For example, a picture may depict two persons in two of its regions both of whom Picture Recognizer **350** can detect simultaneously. In further aspects, where persons, objects, and/or their activities span multiple pictures, Picture Recognizer **350** may detect or recognize persons, objects, and/or their activities by applying the aforementioned pixel or feature comparisons and/or other detection or recognition techniques over a stream of digital pictures (i.e. motion picture, video, etc.). For example, once a person is detected in a digital picture (i.e. frame, etc.) of a stream of digital pictures (i.e. motion picture, video, etc.), the region of pixels comprising the detected person or the person's features can be searched in other pictures of the stream of digital pictures, thereby tracking the person through the stream of digital pictures. In further aspects, Picture Recognizer **350** may detect or recognize a person's activities by identifying and/or analyzing differences between a detected region of pixels of one picture (i.e. frame, etc.) and detected regions of pixels of other pictures in a stream of digital pictures. For example, a region of pixels comprising a person's face can be detected in multiple consecutive pictures of a stream of digital pictures (i.e. motion picture, video, etc.). Differences among the detected regions of the consecutive pictures may be identified in the mouth part of the person's face to indicate smiling or speaking activity. Any technique for recognizing speech from mouth/lip movements can be used in this and other examples. In further aspects, Picture Recognizer **350** may detect or recognize persons, objects, and/or their activities using one or more artificial neural networks, which may include statistical techniques. Examples of artificial neural networks that can be used in Picture Recognizer **350** include convolutional neural networks (CNNs), time delay neural networks (TDNNs), deep neural networks, and/or others. In one example, picture recognition techniques and/or tools involving convolutional neural networks may include identifying and/or analyzing tiled and/or overlapping regions or features of a digital picture, which may then be used to search for pictures with matching regions or features. In another example, features of different convolutional neural networks responsible for spatial and temporal streams can be fused to detect persons, objects, and/or their activities in streams of digital pictures (i.e. motion pictures, videos, etc.). In general, Picture Recognizer **350** may include any machine learning, deep learning, and/or other artificial intelligence techniques. Any other techniques known in art can be utilized in Picture Recognizer **350**. For example, thresholds for similarity, statistical, and/or optimization techniques can be utilized to determine a match in any of the above-described detection or recognition techniques. Picture Recognizer **350** comprises any features, functionalities, and embodiments of Similarity Comparison **125** (later described).

In some exemplary embodiments, object recognition techniques and/or tools such as OpenCV (Open Source Computer Vision) library, CamFind API, Kooaba, 6px API, Dextro API, and/or others can be utilized for detecting or recognizing objects (i.e. objects, animals, people, etc.) in digital pictures. In some aspects, object recognition techniques and/or tools involve identifying and/or analyzing object features such as lines, edges, ridges, corners, blobs, regions, and/or their relative positions, sizes, shapes, etc., which may then be used to search for pictures with matching features. For example, OpenCV library can detect an object (i.e. car, pedestrian, door, building, animal, person, etc.) in one or more digital pictures captured by Picture Capturing Apparatus **90** or stored in an electronic repository, which can then be utilized in VSADO Unit **100**, Artificial Intelligence Unit **110**, and/or other elements.

In other exemplary embodiments, facial recognition techniques and/or tools such as OpenCV (Open Source Computer Vision) library, Animetrics FaceR API, Lambda Labs Facial Recognition API, Face++ SDK, Neven Vision (also known as N-Vision) Engine, and/or others can be utilized for detecting or recognizing faces in digital pictures. In some aspects, facial recognition techniques and/or tools involve identifying and/or analyzing facial features such as the relative position, size, and/or shape of the eyes, nose, cheekbones, jaw, etc., which may then be used to search for pictures with matching features. For example, FaceR API can detect a person's face in one or more digital pictures captured by Picture Capturing Apparatus **90** or stored in an electronic repository, which can then be utilized in VSADO Unit **100**, Artificial Intelligence Unit **110**, and/or other elements.

Referring to FIG. **11**, an embodiment of Artificial Intelligence Unit **110** is illustrated. Artificial Intelligence Unit **110** comprises interconnected Knowledge Structuring Unit **520**, Knowledgebase **530**, Decision-making Unit **540**, and Confirmation Unit **550**. Other additional elements can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate embodiments.

Artificial Intelligence Unit **110** comprises the functionality for learning Device's **98** operation in various visual surroundings. Artificial Intelligence Unit **110** comprises the functionality for learning one or more digital pictures correlated with any instruction sets, data, and/or other information. In some aspects, Artificial Intelligence Unit **110**

comprises the functionality for learning one or more Digital Pictures **525** of Device's **98** surrounding correlated with any Instruction Sets **526** and/or Extra Info **527**. In other aspects, Artificial Intelligence Unit **110** comprises the functionality for learning one or more Digital Pictures **525** of Device's **98** surrounding some of which may not be correlated with any Instruction Sets **526** and/or Extra Info **527**. Further, Artificial Intelligence Unit **110** comprises the functionality for anticipating Device's **98** operation in various visual surroundings. Artificial Intelligence Unit **110** comprises the functionality for anticipating one or more instruction sets, data, and/or other information. Artificial Intelligence Unit **110** comprises the functionality for anticipating one or more Instruction Sets **526** based on one or more incoming Digital Pictures **525** of Device's **98** surrounding. Artificial Intelligence Unit **110** comprises the functionality for anticipating one or more Instruction Sets **526** to be used or executed in Device's **98** autonomous operation. Artificial Intelligence Unit **110** also comprises other disclosed functionalities.

Knowledge Structuring Unit **520**, Knowledgebase **530**, and Decision-making Unit **540** are described later.

Confirmation Unit **550** comprises the functionality for confirming, modifying, evaluating (i.e. rating, etc.), and/or canceling one or more anticipatory Instruction Sets **526**, and/or other functionalities. Confirmation Unit **550** is an optional element that can be omitted depending on implementation. In some embodiments, Confirmation Unit **550** can serve as a means of confirming anticipatory Instruction Sets **526**. For example, Decision-making Unit **540** may determine one or more anticipatory Instruction Sets **526** and provide them to User **50** for confirmation. User **50** may be provided with an interface (i.e. graphical user interface, selectable list of anticipatory Instruction Sets **526**, etc.) to approve or confirm execution of the anticipatory Instruction Sets **526**. In some aspects, Confirmation Unit **550** can automate User **50** confirmation. In one example, if one or more incoming Digital Pictures **525** from Picture Capturing Apparatus **90** and one or more Digital Pictures **525** from a Knowledge Cell **800** were found to be a perfect or highly similar match, anticipatory Instruction Sets **526** correlated with the one or more Digital Pictures **525** from the Knowledge Cell **800** can be automatically executed without User's **50** confirmation. Conversely, if one or more incoming Digital Pictures **525** from Picture Capturing Apparatus **90** and one or more Digital Pictures **525** from a Knowledge Cell **800** were found to be less than a highly similar match, anticipatory Instruction Sets **526** correlated with the one or more Digital Pictures **525** from the Knowledge Cell **800** can be presented to User **50** for confirmation and/or modifying. A threshold that defines a highly or otherwise similar match can be utilized in such implementations. Such threshold can be defined by a user, by VSADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. In other embodiments, Confirmation Unit **550** can serve as a means of modifying or editing anticipatory Instruction Sets **526**. For example, Decision-making Unit **540** may determine one or more anticipatory Instruction Sets **526** and provide them to User **50** for modification. User **50** may be provided with an interface (i.e. graphical user interface, etc.) to modify the anticipatory Instruction Sets **526** before their execution. In further embodiments, Confirmation Unit **550** can serve as a means of evaluating or rating anticipatory Instruction Sets **526** if they matched User's **50** intended operation. For example, Decision-making Unit **540** may determine one or more anticipatory Instruction Sets **526**, which the system may automatically execute. User **50** may be provided with an interface (i.e. graphical user interface, etc.) to rate (i.e. on a scale from 0 to 1, etc.) how well Decision-making Unit **540** predicted the executed anticipatory Instruction Sets **526**. In some aspects, rating can be automatic and based on a particular function or method that rates how well the anticipatory Instruction Sets **526** matched the desired operation. In one example, a rating function or method can assign a higher rating to anticipatory Instruction Sets **526** that were least modified in the confirmation process. In another example, a rating function or method can assign a higher rating to anticipatory Instruction Sets **526** that were canceled least number of times by User **50**. Any other automatic rating function or method can be utilized. In yet other embodiments, Confirmation Unit **550** can serve as a means of canceling anticipatory Instruction Sets **526** if they did not match User's **50** intended operation. For example, Decision-making Unit **540** may determine one or more anticipatory Instruction Sets **526**, which the system may automatically execute. The system may save the state of Device **98**, Processor **11** (save its register values, etc.), Logic Circuit **250**, Application Program **18** (i.e. save its variables, data structures, objects, location of its current instruction, etc.), and/or other processing elements before executing anticipatory Instruction Sets **526**. User **50** may be provided with an interface (i.e. graphical user interface, selectable list of prior executed anticipatory Instruction Sets **526**, etc.) to cancel one or more of the prior executed anticipatory Instruction Sets **526**, and restore Device **98**, Processor **11**, Logic Circuit **250**, Application Program **18**, and/or other processing elements to a prior state. In some aspects, Confirmation Unit **550** can optionally be disabled or omitted in order to provide an uninterrupted operation of Device **98**, Processor **11**, Logic Circuit **250**, and/or Application Program **18**. For example, a microwave oven may be suitable for implementing the user confirmation step, whereas, a robot or vehicle may be less suitable for implementing such interrupting step due to the real time nature of robot or vehicle operation.

Referring to FIG. **12**, an embodiment of Knowledge Structuring Unit **520** correlating individual Digital Pictures **525** with any Instruction Sets **526** and/or Extra Info **527** is illustrated. Knowledge Structuring Unit **520** comprises the functionality for structuring the knowledge of a device's operation in various visual surroundings, and/or other functionalities. Knowledge Structuring Unit **520** comprises the functionality for correlating one or more Digital Pictures **525** of Device's **98** surrounding with any Instruction Sets **526** and/or Extra Info **527**. Knowledge Structuring Unit **520** comprises the functionality for creating or generating Knowledge Cell **800** and storing one or more Digital Pictures **525** correlated with any Instruction Sets **526** and/or Extra Info **527** into the Knowledge Cell **800**. As such, Knowledge Cell **800** comprises the functionality for storing one or more Digital Pictures **525** correlated with any Instruction Sets **526** and/or Extra Info **527**. Knowledge Cell **800** includes a unit of knowledge of how Device **98** operated in a visual surrounding. Once created or generated, Knowledge Cells **800** can be used in/as neurons, nodes, vertices, or other elements in any of the data structures or arrangements (i.e. neural networks, graphs, sequences, etc.) used for storing the knowledge of Device's **98** operation in various visual surroundings, thereby facilitating learning functionalities herein. It should be noted that Extra Info **527** may be optionally used in some implementations to enable enhanced comparisons or decision making in autonomous device operation where applicable, and that Extra Info **527** can be omitted in alternate implementations.

In some embodiments, Knowledge Structuring Unit **520** receives one or more Digital Pictures **525** from Picture Capturing Apparatus **90**. Knowledge Structuring Unit **520** may also receive one or more Instruction Sets **526** from Acquisition Interface **120**. Knowledge Structuring Unit **520** may further receive any Extra Info **527**. Although, Extra Info **527** is not shown in this and/or other figures for clarity of illustration, it should be noted that any Digital Picture **525**, Instruction Set **526**, and/or other element may include or be associated with Extra Info **527**. Knowledge Structuring Unit **520** may correlate one or more Digital Pictures **525** with any Instruction Sets **526** and/or Extra Info **527**. Knowledge Structuring Unit **520** may then create Knowledge Cell **800** and store the one or more Digital Pictures **525** correlated with Instruction Sets **526** and/or Extra Info **527** into the Knowledge Cell **800**. Knowledge Cell **800** may include any data structure or arrangement that can facilitate such storing. For example, Knowledge Structuring Unit **520** may create Knowledge Cell **800***ax* and structure within it Digital Picture **525***a***1** correlated with Instruction Sets **526***a***1**-**526***a***3** and/or any Extra Info **527** (not shown). Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800***ax* a Digital Picture **525***a***2** correlated with Instruction Set **526***a***4** and/or any Extra Info **527** (not shown). Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800***ax* a Digital Picture **525***a***3** without a correlated Instruction Set **526** and/or Extra Info **527**. Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800***ax* a Digital Picture **525***a***4** correlated with Instruction Sets **526***a***5**-**526***a***6** and/or any Extra Info **527** (not shown). Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800***ax* a Digital Picture **525***a***5** without a correlated Instruction Set **526** and/or Extra Info **527**. Knowledge Structuring Unit **520** may structure within Knowledge Cell **800***ax* additional Digital Pictures **525** correlated with any number (including zero [i.e. uncorrelated]) of Instruction Sets **526** and/or Extra Info **527** by following the same logic as described above.

In some embodiments, Knowledge Structuring Unit **520** may correlate a Digital Picture **525** with one or more temporally corresponding Instruction Sets **526** and/or Extra Info **527**. This way, Knowledge Structuring Unit **520** can structure the knowledge of Device's **98** operation at or around the time of the capturing of Digital Pictures **525** of Device's **98** surrounding. Such functionality enables spontaneous or seamless learning of Device's **98** operation in various visual surroundings as user operates the device in real life situations. In some designs, Knowledge Structuring Unit **520** may receive a stream of Instruction Sets **526** used or executed to effect Device's **98** operations as well as a stream of Digital Pictures **525** of Device's **98** surrounding as the operations are performed. Knowledge Structuring Unit **520** can then correlate Digital Pictures **525** from the stream of Digital Pictures **525** with temporally corresponding Instruction Sets **526** from the stream of Instruction Sets **526** and/or any Extra Info **527**. Digital Pictures **525** without a temporally corresponding Instruction Set **526** may be uncorrelated, for instance. In some aspects, Instruction Sets **526** and/or Extra Info **527** that temporally correspond to a Digital Picture **525** may include Instruction Sets **526** used and/or Extra Info **527** obtained at the time of capturing the Digital Picture **525**. In other aspects, Instruction Sets **526** and/or Extra Info **527** that temporally correspond to a Digital Picture **525** may include Instruction Sets **526** used and/or Extra Info **527** obtained within a certain time period before and/or after capturing the Digital Picture **525**. For example, Instruction Sets **526** and/or Extra Info **527** that temporally correspond to a Digital Picture **525** may include Instruction Sets **526** used and/or Extra Info **527** obtained within 50 milliseconds, 1 second, 3 seconds, 20 seconds, 1 minute, 41 minutes, 2 hours, or any other time period before and/or after capturing the Digital Picture **525**. Such time periods can be defined by a user, by VSADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. In other aspects, Instruction Sets **526** and/or Extra Info **527** that temporally correspond to a Digital Picture **525** may include Instruction Sets **526** used and/or Extra Info **527** obtained from the time of capturing of the Digital Picture **525** to the time of capturing of a next Digital Picture **525**. In further aspects, Instruction Sets **526** and/or Extra Info **527** that temporally correspond to a Digital Picture **525** may include Instruction Sets **526** used and/or Extra Info **527** obtained from the time of capturing of a previous Digital Picture **525** to the time of capturing of the Digital Picture **525**. Any other temporal relationship or correspondence between Digital Pictures **525** and correlated Instruction Sets **526** and/or Extra Info **527** can be implemented.

In some embodiments, Knowledge Structuring Unit **520** can structure the knowledge of Device's **98** operation in a visual surrounding into any number of Knowledge Cells **800**. In some aspects, Knowledge Structuring Unit **520** can structure into a Knowledge Cell **800** a single Digital Picture **525** correlated with any Instruction Sets **526** and/or Extra Info **527**. In other aspects, Knowledge Structuring Unit **520** can structure into a Knowledge Cell **800** any number (i.e. 2, 3, 5, 8, 19, 33, 99, 1715, 21822, 393477, 6122805, etc.) of Digital Pictures **525** correlated with any Instruction Sets **526** and/or Extra Info **527**. In a special case, Knowledge Structuring Unit **520** can structure all Digital Pictures **525** correlated with any Instruction Sets **526** and/or Extra Info **527** into a single long Knowledge Cell **800**. In further aspects, Knowledge Structuring Unit **520** can structure Digital Pictures **525** correlated with any Instruction Sets **526** and/or Extra Info **527** into a plurality of Knowledge Cells **800**. In a special case, Knowledge Structuring Unit **520** can store periodic streams of Digital Pictures **525** correlated with any Instruction Sets **526** and/or Extra Info **527** into a plurality of Knowledge Cells **800** such as hourly, daily, weekly, monthly, yearly, or other periodic Knowledge Cells **800**.

In some embodiments, Knowledge Structuring Unit **520** may be responsive to a triggering object, action, event, time, and/or other stimulus. In some aspects, the system can detect or recognize an object in Device's **98** visual surrounding, and Knowledge Structuring Unit **520** can structure into a Knowledge Cell **800** one or more Digital Pictures **525** correlated with any Instruction Sets **526** and/or Extra Info **527** related to the object. For example, Knowledge Structuring Unit **520** can structure into a Knowledge Cell **800** one or more Digital Pictures **525** of a pizza from a microwave oven (i.e. Device **98**, etc.) correlated with any Instruction Sets **526** (i.e. inputs, outputs, or states of the microwave oven's microcontroller, etc.) causing the microwave oven to bake the pizza. Knowledge Structuring Unit **520** can also structure into the Knowledge Cell **800** any Extra Info **527** (i.e. time, location, computed, observed, sensory, and/or other information, etc.). In other aspects, the system can detect or recognize a specific action or operation performed by Device **98**, and Knowledge Structuring Unit **520** can structure into a Knowledge Cell **800** one or more Digital Pictures **525** correlated with any Instruction Sets **526** and/or Extra Info **527** related to the action or operation. For example, Knowledge Structuring Unit **520** can structure into a Knowledge Cell **800** one or more Digital Pictures **525**

depicting screwing of a screw by a robotic arm (i.e. Device **98**, etc.) correlated with any Instruction Sets **526** causing the robotic arm to screw the screw. Knowledge Structuring Unit **520** can also structure into the Knowledge Cell **800** any Extra Info **527** (i.e. time, location, computed, observed, sensory, and/or other information, etc.). In further aspects, the system can detect a person in Device's **98** visual surrounding, and Knowledge Structuring Unit **520** can structure into a Knowledge Cell **800** one or more Digital Pictures **525** correlated with any Instruction Sets **526** and/or Extra Info **527** related to the person. For example, Knowledge Structuring Unit **520** can structure into a Knowledge Cell **800** one or more Digital Pictures **525** of a pedestrian in front of a vehicle (i.e. Device **98**, etc.) correlated with any Instruction Sets **526** causing the vehicle to stop. Knowledge Structuring Unit **520** can also structure into the Knowledge Cell **800** any Extra Info **527** (i.e. time, location, computed, observed, sensory, and/or other information, etc.). In further aspects, the system can detect or recognize a significant change in Device's **98** visual surrounding, and Knowledge Structuring Unit **520** can structure into a Knowledge Cell **800** one or more Digital Pictures **525** correlated with any Instruction Sets **526** and/or Extra Info **527** related to the change in visual surrounding. For example, the system can detect a vehicle's (i.e. Device **98**, etc.) changing direction (i.e. turning left, right, etc.) and Knowledge Structuring Unit **520** can structure into a Knowledge Cell **800** one or more Digital Pictures **525** correlated with any Instruction Sets **526** causing the change of direction. Knowledge Structuring Unit **520** can also structure into the Knowledge Cell **800** any Extra Info **527** (i.e. time, location, computed, observed, sensory, and/or other information, etc.). A vehicle's changing direction may be detected as a significant change in the vehicle's visual surrounding as the view of the vehicle's scenery changes significantly. Any features, functionalities, and embodiments of Picture Recognizer **350** can be utilized in the aforementioned detecting or recognizing. In general, Knowledge Structuring Unit **520** can structure into a Knowledge Cell **800** any Digital Pictures **525** correlated with any Instruction Sets **526** and/or Extra Info **527** related to any triggering object, action, event, time, and/or other stimulus.

In some embodiments, Device **98** may include a plurality of Picture Capturing Apparatuses **90**. In one example, different Picture Capturing Apparatuses **90** may capture Digital Pictures **525** of different angles or sides of Device **98**. In another example, different Picture Capturing Apparatuses **90** may be placed on different sub-devices, sub-systems, or elements of Device **98**. Using multiple Picture Capturing Apparatuses **90** may provide additional visual detail in learning and/or using Device's **98** surrounding for autonomous Device **98** operation. In some designs where multiple Picture Capturing Apparatuses **90** are utilized, multiple VSADO Units **100** can also be utilized (i.e. one VSADO Unit **100** for each Picture Capturing Apparatus **90**, etc.). Digital Pictures **525** of Device's **98** surrounding can be correlated with any Instruction Sets **526** and/or Extra Info **527** as previously described. In other designs where multiple Picture Capturing Apparatuses **90** are utilized, collective Digital Pictures **525** of Device's **98** surrounding from multiple Picture Capturing Apparatuses **90** can be correlated with any Instruction Sets **526** and/or Extra Info **527**.

In some embodiments, Device **98** may include a plurality of Logic Circuits **250**, Processors **11**, Application Programs **18**, and/or other processing elements. For example, each processing element may control a sub-device, sub-system, or an element of Device **98**. Using multiple processing elements may provide enhanced control over Device's **98** operation. In some designs where multiple processing elements are utilized, multiple VSADO Units **100** can also be utilized (i.e. one VSADO Unit **100** for each processing element, etc.). Digital Pictures **525** of Device's **98** surrounding can be correlated with any Instruction Sets **526** and/or Extra Info **527** as previously described. In other designs where multiple processing elements are utilized, Digital Pictures **525** of Device's **98** surrounding can be correlated with any collective Instruction Sets **526** and/or Extra Info **527** used or executed by a plurality of processing elements.

Any combination of the aforementioned multiple Picture Capturing Apparatuses **90**, multiple processing elements, and/or other elements can be implemented in alternate embodiments.

Referring to FIG. **13**, another embodiment of Knowledge Structuring Unit **520** correlating individual Digital Pictures **525** with any Instruction Sets **526** and/or Extra Info **527** is illustrated. In such embodiments, Knowledge Structuring Unit **520** may generate Knowledge Cells **800** each comprising a single Digital Picture **525** correlated with any Instruction Sets **526** and/or Extra Info **527**.

Referring to FIG. **14**, an embodiment of Knowledge Structuring Unit **520** correlating streams of Digital Pictures **525** with any Instruction Sets **526** and/or Extra Info **527** is illustrated. In some aspects, a stream of Digital Pictures **525** may include a collection, a group, a sequence, or other plurality of Digital Pictures **525**. In other aspects, a stream of Digital Pictures **525** may include one or more Digital Pictures **525**. In further aspects, a stream of Digital Pictures **525** may include a digital motion picture (i.e. digital video, etc.) or portion thereof. For example, Knowledge Structuring Unit **520** may create Knowledge Cell **800*ax*** and structure within it a stream of Digital Pictures **525*a*1-525*an*** correlated with Instruction Set **526*a*1** and/or any Extra Info **527** (not shown). Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800*ax*** a stream of Digital Pictures **525*b*1-525*bn*** correlated with Instruction Sets **526*a*2-526*a*4** and/or and Extra Info **527** (not shown). Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800*ax*** a stream of Digital Pictures **525*c*1-525*cn*** without correlated Instruction Sets **526** and/or Extra Info **527**. Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800*ax*** a stream of Digital Pictures **525*d*1-525*dn*** correlated with Instruction Sets **526*a*5-526*a*6** and/or any Extra Info **527** (not shown). Knowledge Structuring Unit **520** may further structure within Knowledge Cell **800*ax*** additional streams of Digital Pictures **525** correlated with any number (including zero [i.e. uncorrelated]) of Instruction Sets **526** and/or Extra Info **527** by following the same logic as described above. The number of Digital Pictures **525** in some or all streams of Digital Pictures **525*a*1-525*an***, **525*b*1-525*bn***, etc. may be equal or different. It should be noted that n or other such letters or indicia may follow the sequence and/or context where they are indicated. Also, a same letter or indicia such as n may represent a different number in different elements of a drawing.

Referring to FIG. **15**, another embodiment of Knowledge Structuring Unit **520** correlating streams of Digital Pictures **525** with any Instruction Sets **526** and/or Extra Info **527** is illustrated. In such embodiments, Knowledge Structuring Unit **520** may generate Knowledge Cells **800** each comprising a single stream of Digital Pictures **525** correlated with any Instruction Sets **526** and/or Extra Info **527**.

Knowledgebase **530** comprises the functionality for storing the knowledge of a device's operation in various visual surroundings, and/or other functionalities. Knowledgebase

530 comprises the functionality for storing one or more Digital Pictures **525** of Device's **98** surrounding correlated with any Instruction Sets **526** and/or Extra Info **527**. Knowledgebase **530** comprises the functionality for storing one or more Knowledge Cells **800** each including one or more Digital Pictures **525** of Device's **98** surrounding correlated with any Instruction Sets **526** and/or Extra Info **527**. In some aspects, Digital Pictures **525** correlated with Instruction Sets **526** and/or Extra Info **527** can be stored directly within Knowledgebase **530** without using Knowledge Cells **800** as the intermediary data structures. In some embodiments, Knowledgebase **530** may be or include Neural Network **530***a* (later described). In other embodiments, Knowledgebase **530** may be or include Graph **530***b* (later described). In further embodiments, Knowledgebase **530** may be or include Collection of Sequences **530***c* (later described). In further embodiments, Knowledgebase **530** may be or include Sequence **533** (later described). In further embodiments, Knowledgebase **530** may be or include Collection of Knowledge Cells **530***d* (later described). In general, Knowledgebase **530** may be or include any data structure or arrangement capable of storing the knowledge of a device's operation in various visual surroundings. Knowledgebase **530** may reside locally on Device **98**, or remotely (i.e. remote Knowledgebase **530**, etc.) on a remote computing device (i.e. server, cloud, etc.) accessible over a network.

Knowledgebase **530** from one Device **98** or VSADO Unit **100** can be transferred to one or more other Devices **98** or VSADO Units **100**. Therefore, the knowledge of Device's **98** operation in various visual surroundings learned on one Device **98** or VSADO Unit **100** can be transferred to one or more other Devices **98** or VSADO Units **100**. In one example, Knowledgebase **530** can be copied or downloaded to a file or other repository from one Device **98** or VSADO Unit **100** and loaded or inserted into another Device **98** or VSADO Unit **100**. In another example, Knowledgebase **530** from one Device **98** or VSADO Unit **100** can be available on a server accessible by other Devices **98** or VSADO Units **100** over a network. Once loaded into or accessed by a receiving Device **98** or VSADO Unit **100**, the receiving Device **98** or VSADO Unit **100** can then implement the knowledge of Device's **98** operation in various visual surroundings learned on the originating Device **98** or VSADO Unit **100**. This functionality enables User **50** such as a professional Device **98** operator to record his/her knowledge, methodology, or style of operating Device **98** in various visual surroundings and/or sell his/her knowledge to other users.

Referring to FIG. **16**, the disclosed artificially intelligent systems, devices, and methods for learning and/or using visual surrounding for autonomous device operation may include various artificial intelligence models and/or techniques. The disclosed systems, devices, and methods are independent of the artificial intelligence model and/or technique used and any model and/or technique can be utilized to facilitate the functionalities described herein. Examples of these models and/or techniques include deep learning, supervised learning, unsupervised learning, neural networks (i.e. convolutional neural network, recurrent neural network, deep neural network, etc.), search-based, logic and/or fuzzy logic-based, optimization-based, tree/graph/other data structure-based, hierarchical, symbolic and/or sub-symbolic, evolutionary, genetic, multi-agent, deterministic, probabilistic, statistical, and/or other models and/or techniques.

In one example shown in Model A, the disclosed artificially intelligent systems, devices, and methods for learning and/or using visual surrounding for autonomous device operation may include a neural network (also referred to as artificial neural network, etc.). As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include a network of Nodes **852** (also referred to as neurons, etc.) and Connections **853** similar to that of a brain. Node **852** can store any data, object, data structure, and/or other item, or reference thereto. Node **852** may also include a function for transforming or manipulating any data, object, data structure, and/or other item. Examples of such transformation functions include mathematical functions (i.e. addition, subtraction, multiplication, division, sin, cos, log, derivative, integral, etc.), object manipulation functions (i.e. creating an object, modifying an object, deleting an object, appending objects, etc.), data structure manipulation functions (i.e. creating a data structure, modifying a data structure, deleting a data structure, creating a data field, modifying a data field, deleting a data field, etc.), and/or other transformation functions. Connection **853** may include or be associated with a value such as a symbolic label or numeric attribute (i.e. weight, cost, capacity, length, etc.). A computational model can be utilized to compute values from inputs based on a pre-programmed or learned function or method. For example, a neural network may include one or more input neurons that can be activated by inputs. Activations of these neurons can then be passed on, weighted, and transformed by a function to other neurons. Neural networks may range from those with only one layer of single direction logic to multi-layer of multi-directional feedback loops. A neural network can use weights to change the parameters of the network's throughput. A neural network can learn by input from its environment or from self-teaching using written-in rules. A neural network can be utilized as a predictive modeling approach in machine learning. An exemplary embodiment of a neural network (i.e. Neural Network **530***a*, etc.) is described later.

In another example shown in Model B, the disclosed artificially intelligent systems, devices, and methods for learning and/or using visual surrounding for autonomous device operation may include a graph or graph-like data structure. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include Nodes **852** (also referred to as vertices or points, etc.) and Connections **853** (also referred to as edges, arrows, lines, arcs, etc.) organized as a graph. In general, any Node **852** in a graph can be connected to any other Node **852**. A Connection **853** may include unordered pair of Nodes **852** in an undirected graph or ordered pair of Nodes **852** in a directed graph. Nodes **852** can be part of the graph structure or external entities represented by indices or references. A graph can be utilized as a predictive modeling approach in machine learning. Nodes **852**, Connections **853**, and/or other elements or operations of a graph may include any features, functionalities, and embodiments of the aforementioned Nodes **852**, Connections **853**, and/or other elements or operations of a neural network, and vice versa. An exemplary embodiment of a graph (i.e. Graph **530***b*, etc.) is described later.

In a further example shown in Model C, the disclosed artificially intelligent systems, devices, and methods for learning and/or using visual surrounding for autonomous device operation may include a tree or tree-like data structure. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include Nodes **852** and Connections **853** (also referred to as refer-

ences, edges, etc.) organized as a tree. In general, a Node **852** in a tree can be connected to any number (i.e. including zero, etc.) of children Nodes **852**. A tree can be utilized as a predictive modeling approach in machine learning. Nodes **852**, Connections **853**, and/or other elements or operations of a tree may include any features, functionalities, and embodiments of the aforementioned Nodes **852**, Connections **853**, and/or other elements or operations of a neural network and/or graph, and vice versa.

In a further example shown in Model D, the disclosed artificially intelligent systems, devices, and methods for learning and/or using visual surrounding for autonomous device operation may include a sequence or sequence-like data structure. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include a structure of Nodes **852** and/or Connections **853** organized as a sequence. In some aspects, Connections **853** may be optionally omitted from a sequence as the sequential order of Nodes **852** in a sequence may be implied in the structure. A sequence can be utilized as a predictive modeling approach in machine learning. Nodes **852**, Connections **853**, and/or other elements or operations of a sequence may include any features, functionalities, and embodiments of the aforementioned Nodes **852**, Connections **853**, and/or other elements or operations of a neural network, graph, and/or tree, and vice versa. An exemplary embodiment of a sequence (i.e. Collection of Sequences **530***c*, Sequence **533**, etc.) is described later.

In yet another example, the disclosed artificially intelligent systems, devices, and methods for learning and/or using visual surrounding for autonomous device operation may include a search-based model and/or technique. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include searching through a collection of possible solutions. For example, a search method can search through a neural network, graph, tree, sequence, or other data structure that includes data elements of interest. A search may use heuristics to limit the search for solutions by eliminating choices that are unlikely to lead to the goal. Heuristic techniques may provide a best guess solution. A search can also include optimization. For example, a search may begin with a guess and then refine the guess incrementally until no more refinements can be made. In a further example, the disclosed systems, devices, and methods may include logic-based model and/or technique. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities can use formal or other type of logic. Logic based models may involve making inferences or deriving conclusions from a set of premises. As such, a logic based system can extend existing knowledge or create new knowledge automatically using inferences. Examples of the types of logic that can be utilized include propositional or sentential logic that comprises logic of statements which can be true or false; first-order logic that allows the use of quantifiers and predicates and that can express facts about objects, their properties, and their relations with each other; fuzzy logic that allows degrees of truth to be represented as a value between 0 and 1 rather than simply 0 (false) or 1 (true), which can be used for uncertain reasoning; subjective logic that comprises a type of probabilistic logic that may take uncertainty and belief into account, which can be suitable for modeling and analyzing situations involving uncertainty, incomplete knowledge and different world views; and/or other types of logic. In a further example, the disclosed systems, devices, and methods may include a probabilistic model and/or technique. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities can be implemented to operate with incomplete or uncertain information where probabilities may affect outcomes. Bayesian network, among other models, is an example of a probabilistic tool used for purposes such as reasoning, learning, planning, perception, and/or others. One of ordinary skill in art will understand that the aforementioned artificial intelligence models and/or techniques are described merely as examples of a variety of possible implementations, and that while all possible artificial intelligence models and/or techniques are too voluminous to describe, other artificial intelligence models and/or techniques known in art are within the scope of this disclosure. One of ordinary skill in art will also recognize that an intelligent system may solve a specific problem by using any model and/or technique that works such as, for example, some systems can be symbolic and logical, some can be sub-symbolic neural networks, some can be deterministic or probabilistic, some can be hierarchical, some may include searching techniques, some may include optimization techniques, while others may use other or a combination of models and/or techniques. In general, any artificial intelligence model and/or technique can be utilized that can facilitate the functionalities described herein.

Referring to FIGS. **17**A-**17**C, embodiments of interconnected Knowledge Cells **800** and updating weights of Connections **853** are illustrated. As shown for example in FIG. **17**A, Knowledge Cell **800***za* is connected to Knowledge Cell **800***zb* and Knowledge Cell **800***zc* by Connection **853***z***1** and Connection **853***z***2**, respectively. Each of Connection **853***z***1** and Connection **853***z***2** may include or be associated with occurrence count, weight, and/or other parameter or data. The number of occurrences may track or store the number of observations that a Knowledge Cell **800** was followed by another Knowledge Cell **800** indicating a connection or relationship between them. For example, Knowledge Cell **800***za* was followed by Knowledge Cell **800***zb* 10 times as indicated by the number of occurrences of Connection **853***z***1**. Also, Knowledge Cell **800***za* was followed by Knowledge Cell **800***zc* times as indicated by the number of occurrences of Connection **853***z***2**. The weight of Connection **853***z***1** can be calculated or determined as the number of occurrences of Connection **853***z***1** divided by the sum of occurrences of all connections (i.e. Connection **853***z***1** and Connection **853***z***2**, etc.) originating from Knowledge Cell **800***za*. Therefore, the weight of Connection **853***z***1** can be calculated or determined as 10/(10+15)=0.4, for example. Also, the weight of Connection **853***z***2** can be calculated or determined as 15/(10+15)=0.6, for example. Therefore, the sum of weights of Connection **853***z***1**, Connection **853***z***2**, and/or any other Connections **853** originating from Knowledge Cell **800***za* may equal to 1 or 100%. As shown for example in FIG. **17**B, in the case that Knowledge Cell **800***zd* is inserted and an observation is made that Knowledge Cell **800***zd* follows Knowledge Cell **800***za*, Connection **853***z***3** can be created between Knowledge Cell **800***za* and Knowledge Cell **800***zd*. The occurrence count of Connection **853***z***3** can be set to 1 and weight determined as 1/(10+15+1)= 0.038. The weights of all other connections (i.e. Connection **853***z***1**, Connection **853***z***2**, etc.) originating from Knowledge Cell **800***za* may be updated to account for the creation of Connection **853***z***3**. Therefore, the weight of Connection **853***z***1** can be updated as 10/(10+15+1)=0.385. The weight

of Connection **853**z**2** can also be updated as 15/(10+15+1)=0.577. As shown for example in FIG. **17**C, in the case that an additional occurrence of Connection **853**z**1** is observed (i.e. Knowledge Cell **800**zb followed Knowledge Cell **800**za, etc.), occurrence count of Connection **853**z**1** and weights of all connections (i.e. Connection **853**z**1**, Connection **853**z**2**, and Connection **853**z**3**, etc.) originating from Knowledge Cell **800**za may be updated to account for this observation. The occurrence count of Connection **853**z**1** can be increased by 1 and its weight updated as 11/(11+15+1)=0.407. The weight of Connection **853**z**2** can also be updated as 15/(11+15+1)=0.556. The weight of Connection **853**z**3** can also be updated as 1/(11+15+1)=0.037.

Referring to FIG. **18**, an embodiment of learning Knowledge Cells **800** comprising one or more Digital Pictures **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Collection of Knowledge Cells **530***d* is illustrated. Collection of Knowledge Cells **530***d* comprises the functionality for storing any number of Knowledge Cells **800**. In some aspects, Knowledge Cells **800** may be stored into or applied onto Collection of Knowledge Cells **530***d* in a learning or training process. In effect, Collection of Knowledge Cells **530***d* may store Knowledge Cells **800** that can later be used to enable autonomous Device **98** operation. In some embodiments, Knowledge Structuring Unit **520** structures or generates Knowledge Cells **800** as previously described and the system applies them onto Collection of Knowledge Cells **530***d*, thereby implementing learning Device's **98** operation in various visual surroundings. The term apply or applying may refer to storing, copying, inserting, updating, or other similar action, therefore, these terms may be used interchangeably herein depending on context. The system can perform Similarity Comparisons **125** (later described) of a newly structured Knowledge Cell **800** from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. If a substantially similar Knowledge Cell **800** is not found in Collection of Knowledge Cells **530***d*, the system may insert (i.e. copy, store, etc.) the Knowledge Cell **800** from Knowledge Structuring Unit **520** into Collection of Knowledge Cells **530***d*, for example. On the other hand, if a substantially similar Knowledge Cell **800** is found in Collection of Knowledge Cells **530***d*, the system may optionally omit inserting the Knowledge Cell **800** from Knowledge Structuring Unit **520** as inserting a substantially similar Knowledge Cell **800** may not add much or any additional knowledge to the Collection of Knowledge Cells **530***d*, for example. Also, inserting a substantially similar Knowledge Cell **800** can optionally be omitted to save storage resources and limit the number of Knowledge Cells **800** that may later need to be processed or compared. Any features, functionalities, and embodiments of Similarity Comparison **125**, importance index (later described), similarity index (later described), and/or other disclosed elements can be utilized to facilitate determination of substantial or other similarity and whether to insert a newly structured Knowledge Cell **800** into Collection of Knowledge Cells **530***d*.

For example, the system can perform Similarity Comparisons **125** (later described) of Knowledge Cell **800***ba* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. In the case that a substantially similar match is found between Knowledge Cell **800***ba* and any of the Knowledge Cells **800** in Collection of Knowledge Cells **530***d*, the system may perform no action. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bb* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. In the case that a substantially similar match is not found, the system may insert a new Knowledge Cell **800** into Collection of Knowledge Cells **530***d* and copy Knowledge Cell **800***bb* into the inserted new Knowledge Cell **800**. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bc* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. In the case that a substantially similar match is found between Knowledge Cell **800***bc* and any of the Knowledge Cells **800** in Collection of Knowledge Cells **530***d*, the system may perform no action. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bd* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. In the case that a substantially similar match is not found, the system may insert a new Knowledge Cell **800** into Collection of Knowledge Cells **530***d* and copy Knowledge Cell **800***bd* into the inserted new Knowledge Cell **800**. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***be* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. In the case that a substantially similar match is not found, the system may insert a new Knowledge Cell **800** into Collection of Knowledge Cells **530***d* and copy Knowledge Cell **800***be* into the inserted new Knowledge Cell **800**. Applying any additional Knowledge Cells **800** from Knowledge Structuring Unit **520** onto Collection of Knowledge Cells **530***d* follows similar logic or process as the above-described.

Referring to FIG. **19**, an embodiment of learning Knowledge Cells **800** comprising one or more Digital Pictures **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Neural Network **530***a* is illustrated. Neural Network **530***a* includes a number of neurons or Nodes **852** interconnected by Connections **853** as previously described. Knowledge Cells **800** are shown instead of Nodes **852** to simplify the illustration as Node **852** includes a Knowledge Cell **800**, for example. Therefore, Knowledge Cells **800** and Nodes **852** can be used interchangeably herein depending on context. It should be noted that Node **852** may include other elements and/or functionalities instead of or in addition to Knowledge Cell **800**. In some aspects, Knowledge Cells **800** may be stored into or applied onto Neural Network **530***a* individually or collectively in a learning or training process. In some designs, Neural Network **530***a* comprises a number of Layers **854** each of which may include one or more Knowledge Cells **800**. Knowledge Cells **800** in successive Layers **854** can be connected by Connections **853**. Connection **853** may include or be associated with occurrence count, weight, and/or other parameter or data as previously described. Neural Network **530***a* may include any number of Layers **854** comprising any number of Knowledge Cells **800**. In some aspects, Neural Network **530***a* may store Knowledge Cells **800** interconnected by Connections **853** where following a path through the Neural Network **530***a* can later be used to enable autonomous Device **98** operation. It should be understood that, in some embodiments, Knowledge Cells **800** in one Layer **854** of Neural Network **530***a* need not be connected only with Knowledge Cells **800** in a successive Layer **854**, but also in any other Layer **854**, thereby creating shortcuts (i.e. shortcut Connections **853**, etc.) through Neural Network **530***a*. A Knowledge Cell **800** can also be connected to itself such as, for example, in recurrent neural networks. In general, any Knowledge Cell **800** can be connected with any other Knowledge Cell **800** anywhere else in Neural Network **530***a*. In further embodiments, back-propagation of any data or information can be

implemented. In one example, back-propagation of similarity (i.e. similarity index, etc.) of compared Knowledge Cells **800** in a path through Neural Network **530***a* can be implemented. In another example, back-propagation of errors can be implemented. Such back-propagations can then be used to adjust occurrence counts and/or weights of Connections **853** for better future predictions, for example. Any other back-propagation can be implemented for other purposes. Any combination of Nodes **852** (i.e. Nodes **852** comprising Knowledge Cells **800**, etc.), Connections **853**, Layers **854**, and/or other elements or techniques can be implemented in alternate embodiments. Neural Network **530***a* may include any type or form of a neural network known in art such as a feed-forward neural network, a back-propagating neural network, a recurrent neural network, a convolutional neural network, deep neural network, and/or others including a custom neural network.

In some embodiments, Knowledge Structuring Unit **520** structures or generates Knowledge Cells **800** and the system applies them onto Neural Network **530***a*, thereby implementing learning Device's **98** operation in various visual surroundings. The system can perform Similarity Comparisons **125** (later described) of a Knowledge Cell **800** from Knowledge Structuring Unit **520** with Knowledge Cells **800** in a corresponding Layer **854** of Neural Network **530***a*. If a substantially similar Knowledge Cell **800** is not found in the corresponding Layer **854** of Neural Network **530***a*, the system may insert (i.e. copy, store, etc.) the Knowledge Cell **800** from Knowledge Structuring Unit **520** into the corresponding Layer **854** of Neural Network **530***a*, and create a Connection **853** to the inserted Knowledge Cell **800** from a Knowledge Cell **800** in a prior Layer **854** including assigning an occurrence count to the new Connection **853**, calculating a weight of the new Connection **853**, and updating any other Connections **853** originating from the Knowledge Cell **800** in the prior Layer **854**. On the other hand, if a substantially similar Knowledge Cell **800** is found in the corresponding Layer **854** of Neural Network **530***a*, the system may update occurrence count and weight of Connection **853** to that Knowledge Cell **800** from a Knowledge Cell **800** in a prior Layer **854**, and update any other Connections **853** originating from the Knowledge Cell **800** in the prior Layer **854**.

For example, the system can perform Similarity Comparisons **125** (later described) of Knowledge Cell **800***ba* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Layer **854***a* of Neural Network **530***a*. In the case that a substantially similar match is found between Knowledge Cell **800***ba* and Knowledge Cell **800***ea*, the system may perform no action since Knowledge Cell **800***ea* is the initial Knowledge Cell **800**. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bb* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Layer **854***b* of Neural Network **530***a*. In the case that a substantially similar match is found between Knowledge Cell **800***bb* and Knowledge Cell **800***eb*, the system may update occurrence count and weight of Connection **853***e***1** between Knowledge Cell **800***ea* and Knowledge Cell **800***eb*, and update weights of other Connections **853** originating from Knowledge Cell **800***ea* as previously described. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bc* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Layer **854***c* of Neural Network **530***a*. In the case that a substantially similar match is not found, the system may insert Knowledge Cell **800***ec* into Layer **854***c* and copy Knowledge Cell **800***bc* into the inserted Knowledge Cell **800***ec*. The system may also create Connection **853***e***2** between Knowledge Cell **800***eb* and Knowledge Cell **800***ec* with occurrence count of 1 and weight calculated based on the occurrence count as previously described. The system may also update weights of other Connections **853** (one in this example) originating from Knowledge Cell **800***eb* as previously described. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bd* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Layer **854***d* of Neural Network **530***a*. In the case that a substantially similar match is not found, the system may insert Knowledge Cell **800***ed* into Layer **854***d* and copy Knowledge Cell **800***bd* into the inserted Knowledge Cell **800***ed*. The system may also create Connection **853***e***3** between Knowledge Cell **800***ec* and Knowledge Cell **800***ed* with occurrence count of 1 and weight of 1. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***be* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Layer **854***e* of Neural Network **530***a*. In the case that a substantially similar match is not found, the system may insert Knowledge Cell **800***ee* into Layer **854***e* and copy Knowledge Cell **800***be* into the inserted Knowledge Cell **800***ee*. The system may also create Connection **853***e***4** between Knowledge Cell **800***ed* and Knowledge Cell **800***ee* with occurrence count of 1 and weight of 1. Applying any additional Knowledge Cells **800** from Knowledge Structuring Unit **520** onto Neural Network **530***a* follows similar logic or process as the above-described.

Similarity Comparison **125** comprises the functionality for comparing or matching Knowledge Cells **800** or portions thereof, and/or other functionalities. Similarity Comparison **125** comprises the functionality for comparing or matching Digital Pictures **525** or portions thereof. Similarity Comparison **125** comprises the functionality for comparing or matching streams of Digital Pictures **525** or portions thereof. Similarity Comparison **125** comprises the functionality for comparing or matching Instruction Sets **526**, Extra Info **527**, text (i.e. characters, words, phrases, etc.), pictures, sounds, data, and/or other elements or portions thereof. Similarity Comparison **125** may include functions, rules, and/or logic for performing matching or comparisons and for determining that while a perfect match is not found, a similar or partial match has been found. In some aspects, Similarity Comparison **125** may include determining substantial similarity or substantial match of compared elements. In other aspects, a partial match may include a substantial or otherwise similar match, and vice versa. Although, substantial similarity or substantial match is frequently used herein, it should be understood that any level of similarity, however high or low, may be utilized as defined by the rules (i.e. thresholds, etc.) for similarity. The rules for similarity or similar match can be defined by a user, by VSADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. In some designs, Similarity Comparison **125** comprises the functionality to automatically define appropriately strict rules for determining similarity of the compared elements. Similarity Comparison **125** can therefore set, reset, and/or adjust the strictness of the rules for finding or determining similarity of the compared elements, thereby fine tuning Similarity Comparison **125** so that the rules for determining similarity are appropriately strict. In some aspects, the rules for determining similarity may include a similarity threshold. As such, Similarity Comparison **125** can determine similarity of compared elements if their similarity exceeds a similarity threshold. In other aspects, the rules for determining similarity may

include a difference threshold. As such, Similarity Comparison **125** can determine similarity of compared elements if their difference is lower than a difference threshold. In further aspects, the rules for determining similarity may include other thresholds.

In some embodiments, in determining similarity of Knowledge Cells **800**, Similarity Comparison **125** can compare one or more Digital Pictures **525** or portions (i.e. regions, features, pixels, etc.) thereof from one Knowledge Cell **800** with one or more Digital Pictures **525** or portions thereof from another Knowledge Cell **800**. In some aspects, total equivalence is achieved when all Digital Pictures **525** or portions thereof of the compared Knowledge Cells **800** match. If total equivalence is not found, Similarity Comparison **125** may attempt to determine substantial or other similarity. Any features, functionalities, and embodiments of the previously described Picture Recognizer **350** can be used in determining such substantial similarity.

In some embodiments where compared Knowledge Cells **800** include a single Digital Picture **525**, Similarity Comparison **125** can compare Digital Picture **525** from one Knowledge Cell **800** with Digital Picture **525** from another Knowledge Cell **800** using comparison techniques for individual pictures described below. In some embodiments where compared Knowledge Cells **800** include streams of Digital Pictures **525** (i.e. motion pictures, videos, etc.), Similarity Comparison **125** can compare a stream of Digital Pictures **525** from one Knowledge Cell **800** with a stream of Digital Pictures **525** from another Knowledge Cell **800**. Such comparison may include comparing Digital Pictures **525** from one Knowledge Cell **800** with corresponding (i.e. similarly positioned, temporally related, etc.) Digital Pictures **525** from another Knowledge Cell **800**. In one example, a 67th Digital Picture **525** from one Knowledge Cell **800** can be compared with a 67th Digital Picture **525** from another Knowledge Cell **800**. In another example, a 67th Digital Picture **525** from one Knowledge Cell **800** can be compared with a number of Digital Picture **525** around (i.e. preceding and/or following) a 67th Digital Picture **525** from another Knowledge Cell **800**. This way, flexibility can be implemented in finding a substantially similar Digital Picture **525** if the Digital Pictures **525** in the compared Knowledge Cells **800** are not perfectly aligned. In other aspects, Similarity Comparison **125** can utilize Dynamic Time Warping (DTW) and/or other techniques know in art for comparing and/or aligning temporal sequences (i.e. streams of Digital Pictures **525**, etc.) that may vary in time or speed. Once the corresponding (i.e. similarly positioned, temporally related, time warped/aligned, etc.) Digital Pictures **525** in the compared streams of Digital Pictures **525** are compared and their substantial similarity determined using comparison techniques for individual pictures described below, Similarity Comparison **125** can utilize a threshold for the number or percentage of matching or substantially matching Digital Pictures **525** for determining substantial similarity of the compared Knowledge Cells **800**. In some aspects, substantial similarity can be achieved when most of the Digital Pictures **525** or portions (i.e. regions, features, pixels, etc.) thereof of the compared Knowledge Cells **800** match or substantially match. In other aspects, substantial similarity can be achieved when at least a threshold number or percentage of Digital Pictures **525** or portions thereof of the compared Knowledge Cells **800** match or substantially match. Similarly, substantial similarity can be achieved when a number or percentage of matching or substantially matching Digital Pictures **525** or portions thereof of the compared Knowledge Cells **800** exceeds a threshold. In further aspects, substantial similarity can be achieved when all but a threshold number or percentage of Digital Pictures **525** or portions thereof of the compared Knowledge Cells **800** match or substantially match. Such thresholds can be defined by a user, by VSADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. In one example, substantial similarity can be achieved when at least 1, 2, 3, 4, or any other threshold number of Digital Pictures **525** or portions thereof of the compared Knowledge Cells **800** match or substantially match. Similarly, substantial similarity can be achieved when the number of matching or substantially matching Digital Pictures **525** or portions thereof of the compared Knowledge Cells **800** exceeds 1, 2, 3, 4, or any other threshold number. In another example, substantial similarity can be achieved when at least 10%, 21%, 30%, 49%, 66%, 89%, 93%, or any other percentage of Digital Pictures **525** or portions thereof of the compared Knowledge Cells **800** match or substantially match. Similarly, substantial similarity can be achieved when the percentage of matching or substantially matching Digital Pictures **525** or portions thereof of the compared Knowledge Cells **800** exceeds 10%, 21%, 30%, 49%, 66%, 89%, 93%, or any other threshold percentage. In other embodiments, substantial similarity of the compared Knowledge Cells **800** can be achieved in terms of matches or substantial matches in more important (i.e. as indicated by importance index [later described], etc.) Digital Pictures **525** or portions thereof, thereby tolerating mismatches in less important Digital Pictures **525** or portions thereof. In one example, substantial similarity can be achieved when matches or substantial matches are found with respect to more substantive Digital Pictures **525** (i.e. pictures comprising content of interest [i.e. persons, objects, etc.], etc.) or portions thereof of the compared Knowledge Cells **800**, thereby tolerating mismatches in less substantive Digital Pictures **525** (i.e. pictures comprising background, insignificant content, etc.) or portions thereof. In another example, substantial similarity can be achieved when matches or substantial matches are found in earlier Digital Pictures **525** or portions thereof of the compared Knowledge Cells **800**, thereby tolerating mismatches in later Digital Pictures **525** or portions thereof. In general, any importance or weight can be assigned to any Digital Picture **525** or portion thereof, and/or other elements. In some designs, Similarity Comparison **125** can be configured to omit any Digital Picture **525** or portion thereof from the comparison. In one example, less substantive Digital Pictures **525** or portions thereof can be omitted. In another example, some or all Digital Pictures **525** or portions thereof related to a specific time period can be omitted. In a further example, later Digital Pictures **525** or portions thereof can be omitted. In further embodiments, substantial similarity can be achieved taking into account the number of Digital Pictures **525** of the compared Knowledge Cells **800**. For example, substantial similarity can be achieved if the number, in addition to the content, of Digital Pictures **525** of the compared Knowledge Cells **800** match or substantially match. In further embodiments, substantial similarity can be achieved taking into account the objects detected within Digital Pictures **525** and/or other features of Digital Pictures **525** of the compared Knowledge Cells **800**. For example, substantial similarity can be achieved if same or similar objects are detected in Digital Pictures **525** of the compared Knowledge Cells **800**. Any features, functionalities, and embodiments of Picture Recognizer **350** can be used in such detection. In some aspects, Similarity Comparison **125** can

compare the number, objects detected, and/or other features of Digital Pictures **525** as an initial check before proceeding to further detailed comparisons.

Similarity Comparison **125** can automatically adjust (i.e. increase or decrease) the strictness of the rules for determining substantial similarity of Knowledge Cells **800**. In some aspects, such adjustment in strictness can be done by Similarity Comparison **125** in response to determining that total equivalence of compared Knowledge Cells **800** had not been found. Similarity Comparison **125** can keep adjusting the strictness of the rules until substantially similarity is found. All the rules or settings of substantial similarity can be set, reset, or adjusted by Similarity Comparison **125** in response to another strictness level determination. For example, Similarity Comparison **125** may attempt to find a match or substantial match in a certain percentage (i.e. 95%, etc.) of Digital Pictures **525** or portions thereof from the compared Knowledge Cells **800**. If the comparison does not determine substantial similarity of compared Knowledge Cells **800**, Similarity Comparison **125** may decide to decrease the strictness of the rules. In response, Similarity Comparison **125** may attempt to find fewer matching or substantially matching Digital Pictures **525** or portions thereof than in the previous attempt using stricter rules. If the comparison still does not determine substantial similarity of compared Knowledge Cells **800**, Similarity Comparison **125** may further decrease (i.e. down to a certain minimum strictness or threshold, etc.) the strictness by requiring fewer Digital Pictures **525** or portions thereof to match or substantially match, thereby further increasing a chance of finding substantial similarity in compared Knowledge Cells **800**. In further aspects, an adjustment in strictness can be done by Similarity Comparison **125** in response to determining that multiple substantially similar Knowledge Cells **800** had been found. Similarity Comparison **125** can keep adjusting the strictness of the rules until a best of the substantially similar Knowledge Cells **800** is found. For example, Similarity Comparison **125** may attempt to find a match or substantial match in a certain percentage (i.e. 70%, etc.) of Digital Pictures **525** or portions thereof from the compared Knowledge Cells **800**. If the comparison determines a number of substantially similar Knowledge Cells **800**, Similarity Comparison **125** may decide to increase the strictness of the rules to decrease the number of substantially similar Knowledge Cells **800**. In response, Similarity Comparison **125** may attempt to find more matching or substantially matching Digital Pictures **525** or portions thereof in addition to the earlier found Digital Pictures **525** or portions thereof to limit the number of substantially similar Knowledge Cells **800**. If the comparison still provides more than one substantially similar Knowledge Cell **800**, Similarity Comparison **125** may further increase the strictness by requiring additional Digital Pictures **525** or portions thereof to match or substantially match, thereby further narrowing the number of substantially similar Knowledge Cells **800** until a best substantially similar Knowledge Cells **800** is found.

In some embodiments, in determining substantial similarity of individual Digital Pictures **525** (i.e. Digital Pictures **525** from the compared Knowledge Cells **800**, etc.), Similarity Comparison **125** can compare one or more regions of one Digital Picture **525** with one or more regions of another Digital Picture **525**. A region may include a collection of pixels. In some aspects, a region may include detected or recognized content of interest such as an object or person. Such region may be detected using any features, functionalities, and embodiments of Picture Recognizer **350**. In other aspects, a region may include content defined using a picture segmentation technique. Examples of picture segmentation techniques include thresholding, clustering, region-growing, edge detection, curve propagation, level sets, graph partitioning, model-based segmentation, trainable segmentation (i.e. artificial neural networks, etc.), and/or others. In further aspects, a region may include content defined using any technique. In further aspects, a region may include any arbitrary region comprising any arbitrary content. Once regions of the compared Digital Pictures **525** are known, Similarity Comparison **125** can compare the regions to determine substantial similarity of the compared Digital Pictures **525**. In some aspects, total equivalence is found when all regions of one Digital Picture **525** match all regions of another Digital Picture **525**. In other aspects, if total equivalence is not found, Similarity Comparison **125** may attempt to determine substantial similarity of compared Digital Pictures **525**. In one example, substantial similarity can be achieved when most of the regions of the compared Digital Picture **525** match or substantially match. In another example, substantial similarity can be achieved when at least a threshold number (i.e. 1, 2, 5, 11, 39, etc.) or percentage (i.e. 38%, 63%, 77%, 84%, 98%, etc.) of regions of the compared Digital Pictures **525** match or substantially match. Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching regions of the compared Digital Pictures **525** exceeds a threshold number (i.e. 1, 2, 5, 11, 39, etc.) or a threshold percentage (i.e. 48%, 63%, 77%, 84%, 98%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or percentage of regions of the compared Digital Pictures **525** match or substantially match. Such thresholds can be defined by a user, by VSADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In further aspects, Similarity Comparison **125** can utilize the type of regions for determining substantial similarity of Digital Pictures **525**. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to more substantive, larger, and/or other regions, thereby tolerating mismatches in less substantive, smaller, and/or other regions. In further aspects, Similarity Comparison **125** can utilize the importance (i.e. as indicated by importance index [later described], etc.) of regions for determining substantial similarity of Digital Pictures **525**. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to more important regions such as the above described more substantive, larger, and/or other regions, thereby tolerating mismatches in less important regions such as less substantive, smaller, and/or other regions. In further aspects, Similarity Comparison **125** can omit some of the regions from the comparison in determining substantial similarity of Digital Pictures **525**. In one example, isolated regions can be omitted from comparison. In another example, less substantive or smaller regions can be omitted from comparison. In general, any region can be omitted from comparison. In further aspects, Similarity Comparison **125** can focus on certain regions of interest from the compared Digital Pictures **525**. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to regions comprising persons or parts (i.e. head, arm, leg, etc.) thereof, large objects, close objects, and/or other content of interest, thereby tolerating mismatches in regions comprising the background, insignificant content, and/or other content. In further aspects, Similarity Comparison **125** can detect or

recognize persons or objects in the compared Digital Pictures **525** using regions. Any features, functionalities, and embodiments of Picture Recognizer **350** can be used in such detection or recognition. Once a person or object is detected in a Digital Picture **525**, Similarity Comparison **125** may attempt to detect the person or object in the compared Digital Picture **525**. In one example, substantial similarity can be achieved when the compared Digital Pictures **525** comprise one or more same persons or objects. In another example concerning streams of Digital Pictures **525**, substantial similarity can be achieved when the compared streams of Digital Pictures **525** comprise a detected person or object in at least a threshold number or percentage of their pictures.

Similarity Comparison **125** can automatically adjust (i.e. increase or decrease) the strictness of the rules for determining substantial similarity of Digital Pictures **525** using regions. In some aspects, such adjustment in strictness can be done by Similarity Comparison **125** in response to determining that total equivalence of compared Digital Pictures **525** had not been found. Similarity Comparison **125** can keep adjusting the strictness rules until a substantial similarity is found. All the rules or settings of substantial similarity can be set, reset, or adjusted by Similarity Comparison **125** in response to another strictness level determination. For example, Similarity Comparison **125** may attempt to find a match or substantial match in a certain percentage (i.e. 74%, etc.) of regions from the compared Digital Pictures **525**. If the comparison does not determine substantial similarity of compared Digital Pictures **525**, Similarity Comparison **125** may decide to decrease the strictness of the rules. In response, Similarity Comparison **125** may attempt to find fewer matching or substantially matching regions than in the previous attempt using stricter rules. If the comparison still does not determine substantial similarity of compared Digital Pictures **525**, Similarity Comparison **125** may further decrease the strictness (i.e. down to a certain minimum strictness or threshold, etc.) by requiring fewer regions to match or substantially match, thereby further increasing a chance of finding substantial similarity in compared Digital Pictures **525**.

Where a reference to a region is used herein it should be understood that a portion of a region or a collection of regions can be used instead of or in addition to the region. In one example, instead of or in addition to regions, individual pixels and/or features that constitute a region can be compared. In another example, instead of or in addition to regions, collections of regions can be compared. As such, any operations, rules, logic, and/or functions operating on regions similarly apply to any portion of a region and/or any collection of regions. In general, whole regions, portions of a region, and/or collections of regions, including any operations thereon, can be combined to arrive at desired results. Some or all of the above-described rules, logic, and/or techniques can be utilized alone or in combination with each other or with other rules, logic, and/or techniques. One of ordinary skill in art will recognize that other techniques known in art for determining similarity of digital pictures, streams of digital pictures, and/or other data that would be too voluminous to describe are within the scope of this disclosure.

In some embodiments, in determining substantial similarity of individual Digital Pictures **525** (i.e. Digital Pictures **525** from the compared Knowledge Cells **800**, etc.), Similarity Comparison **125** can compare one or more features of one Digital Picture **525** with one or more features of another Digital Picture **525**. A feature may include a collection of pixels. Some of the steps or elements in a feature oriented technique include pre-processing, feature extraction, detection/segmentation, decision-making, and/or others, or a combination thereof, each of which may include its own sub-steps or sub-elements depending on the application. Examples of features that can be used include lines, edges, ridges, corners, blobs, and/or others. Examples of feature extraction techniques include Canny, Sobe, Kayyali, Harris & Stephens et al, SUSAN, Level Curve Curvature, FAST, Laplacian of Gaussian, Difference of Gaussians, Determinant of Hessian, MSER, PCBR, Grey-level Blobs, and/or others. Once features of the compared Digital Pictures **525** are known, Similarity Comparison **125** can compare the features to determine substantial similarity. In some aspects, total equivalence is found when all features of one Digital Picture **525** match all features of another Digital Picture **525**. In other aspects, if total equivalence is not found, Similarity Comparison **125** may attempt to determine substantial similarity of compared Digital Pictures **525**. In one example, substantial similarity can be achieved when most of the features of the compared Digital Picture **525** match or substantially match. In another example, substantial similarity can be achieved when at least a threshold number (i.e. 3, 22, 47, 93, 128, 431, etc.) or percentage (i.e. 49%, 53%, 68%, 72%, 95%, etc.) of features of the compared Digital Pictures **525** match or substantially match. Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching features of the compared Digital Pictures **525** exceeds a threshold number (i.e. 3, 22, 47, 93, 128, 431, etc.) or a threshold percentage (i.e. 49%, 53%, 68%, 72%, 95%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or percentage of features of the compared Digital Pictures **525** match or substantially match. Such thresholds can be defined by a user, by VSADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In further aspects, Similarity Comparison **125** can utilize the type of features for determining substantial similarity of Digital Pictures **525**. In one example, substantial similarity can be achieved when matches or substantial matches are found with respect to edges, thereby tolerating mismatches in blobs. In another example, substantial similarity can be achieved when matches or substantial matches are found with respect to more substantive, larger, and/or other features, thereby tolerating mismatches in less substantive, smaller, and/or other features. In further aspects, Similarity Comparison **125** can utilize the importance (i.e. as indicated by importance index [later described], etc.) of features for determining substantial similarity of Digital Pictures **525**. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to more important features such as the above described more substantive, larger, and/or other features, thereby tolerating mismatches in less important features such as less substantive, smaller, and/or other features. In further aspects, Similarity Comparison **125** can omit some of the features from the comparison in determining substantial similarity of Digital Pictures **525**. In one example, isolated features can be omitted from comparison. In another example, less substantive or smaller features can be omitted from comparison. In general, any feature can be omitted from comparison. In further aspects, Similarity Comparison **125** can focus on features in certain regions of interest of the compared Digital Pictures **525**. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to features in

regions comprising persons or parts (i.e. head, arm, leg, etc.) thereof, large objects, close objects, and/or other objects, thereby tolerating mismatches in features of regions comprising the background, insignificant content, and/or other regions. In further aspects, Similarity Comparison **125** can detect or recognize persons or objects in the compared Digital Pictures **525**. Any features, functionalities, and embodiments of Picture Recognizer **350** can be used in such detection or recognition. Once a person or object is detected in a Digital Picture **525**, Similarity Comparison **125** may attempt to detect the person or object in the compared Digital Picture **525**. In one example, substantial similarity can be achieved when the compared Digital Pictures **525** comprise one or more same persons or objects. In another example concerning streams of Digital Pictures **525**, substantial similarity can be achieved when the compared streams of Digital Pictures **525** comprise a detected person or object in at least a threshold number or percentage of their pictures. In further aspects, Similarity Comparison **125** may include identifying and/or analyzing tiled and/or overlapping features, which can then be combined (i.e. similar to some process steps in convolutional neural networks, etc.) and compared to determine substantial similarity of Digital Pictures **525**.

Similarity Comparison **125** can automatically adjust (i.e. increase or decrease) the strictness of the rules for determining substantial similarity of Digital Pictures **525** using features. In some aspects, such adjustment in strictness can be done by Similarity Comparison **125** in response to determining that total equivalence of compared Digital Pictures **525** had not been found. Similarity Comparison **125** can keep adjusting the strictness rules until a substantial similarity is found. All the rules or settings of substantial similarity can be set, reset, or adjusted by Similarity Comparison **125** in response to another strictness level determination. For example, Similarity Comparison **125** may attempt to find a match or substantial match in a certain percentage (i.e. 89%, etc.) of features from the compared Digital Pictures **525**. If the comparison does not determine substantial similarity of compared Digital Pictures **525**, Similarity Comparison **125** may decide to decrease the strictness of the rules. In response, Similarity Comparison **125** may attempt to find fewer matching or substantially matching features than in the previous attempt using stricter rules. If the comparison still does not determine substantial similarity of compared Digital Pictures **525**, Similarity Comparison **125** may further decrease the strictness (i.e. down to a certain minimum strictness or threshold, etc.) by requiring fewer features to match or substantially match, thereby further increasing a chance of finding substantial similarity in compared Digital Pictures **525**.

Where a reference to a feature is used herein it should be understood that a portion of a feature or a collection of features can be used instead of or in addition to the feature. In one example, instead of or in addition to features, individual pixels that constitute a feature can be compared. In another example, instead of or in addition to features, collections of features can be compared. In a further example, levels of features where a feature on one level includes one or more features from another level (i.e. prior level, etc.) can be compared. As such, any operations, rules, logic, and/or functions operating on features similarly apply to any portion of a feature and/or any collection of features. In general, whole features, portions of a feature, and/or collections of features, including any operations thereon, can be combined to arrive at desired results. Some or all of the above-described rules, logic, and/or techniques can be utilized alone or in combination with each other or with other rules, logic, and/or techniques. One of ordinary skill in art will recognize that other techniques known in art for determining similarity of digital pictures, streams of digital pictures, and/or other data that would be too voluminous to describe are within the scope of this disclosure.

In some embodiments, in determining substantial similarity of individual Digital Pictures **525** (i.e. Digital Pictures **525** from the compared Knowledge Cells **800**, etc.), Similarity Comparison **125** can compare pixels of one Digital Picture **525** with pixels of another Digital Picture **525**. In some aspects, total equivalence is found when all pixels of one Digital Picture **525** match all pixels of another Digital Picture **525**. In other aspects, if total equivalence is not found, Similarity Comparison **125** may attempt to determine substantial similarity. In one example, substantial similarity can be achieved when most of the pixels from the compared Digital Pictures **525** match or substantially match. In another example, substantial similarity can be achieved when at least a threshold number (i.e. 449, 2219, 92229, 442990, 1000028, etc.) or percentage (i.e. 39%, 45%, 58%, 72%, 92%, etc.) of pixels from the compared Digital Pictures **525** match or substantially match. Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching pixels from the compared Digital Pictures **525** exceeds a threshold number (i.e. 449, 2219, 92229, 442990, 1000028, etc.) or a threshold percentage (i.e. 39%, 45%, 58%, 72%, 92%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or percentage of pixels from the compared Digital Pictures **525** match or substantially match. Such thresholds can be defined by a user, by VSADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In further aspects, Similarity Comparison **125** can omit some of the pixels from the comparison in determining substantial similarity of Digital Pictures **525**. In one example, pixels composing the background or any insignificant content can be omitted from comparison. In general, any pixel can be omitted from comparison. In further aspects, Similarity Comparison **125** can focus on pixels in certain regions of interest in determining substantial similarity of Digital Pictures **525**. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to pixels in regions comprising persons or parts (i.e. head, arm, leg, etc.) thereof, large objects, close objects, and/or other content of interest, thereby tolerating mismatches in pixels in regions comprising the background, insignificant content, and/or other content.

Similarity Comparison **125** can automatically adjust (i.e. increase or decrease) the strictness of the rules for determining substantial similarity of Digital Pictures **525** using pixels. In some aspects, such adjustment in strictness can be done by Similarity Comparison **125** in response to determining that total equivalence of compared Digital Pictures **525** had not been found. Similarity Comparison **125** can keep adjusting the strictness rules until a substantial similarity is found. All the rules or settings of substantial similarity can be set, reset, or adjusted by Similarity Comparison **125** in response to another strictness level determination. For example, Similarity Comparison **125** may attempt to find a match or substantial match in a certain percentage (i.e. 77%, etc.) of pixels from the compared Digital Pictures **525**. If the comparison does not determine substantial similarity of compared Digital Pictures **525**, Similarity Comparison **125** may decide to decrease the

strictness of the rules. In response, Similarity Comparison **125** may attempt to find fewer matching or substantially matching pixels than in the previous attempt using stricter rules. If the comparison still does not determine substantial similarity of compared Digital Pictures **525**, Similarity Comparison **125** may further decrease the strictness (i.e. down to a certain minimum strictness or threshold, etc.) by requiring fewer pixels to match or substantially match, thereby further increasing a chance of finding substantial similarity in compared Digital Pictures **525**.

Where a reference to a pixel is used herein it should be understood that a collection of pixels can be used instead of or in addition to the pixel. For example, instead of or in addition to pixels, collections of pixels can be compared. As such, any operations, rules, logic, and/or functions operating on pixels similarly apply to any collection of pixels. In general, pixels and/or collections of pixels, including any operations thereon, can be combined to arrive at desired results. Some or all of the above-described rules, logic, and/or techniques can be utilized alone or in combination with each other or with other rules, logic, and/or techniques. Any of the previously described features, functionalities, and embodiments of Similarity Comparison **125** for determining substantial similarity of Digital Pictures **525** using regions and/or features can similarly be used for pixels. One of ordinary skill in art will recognize that other techniques known in art for determining similarity of digital pictures, streams of digital pictures, and/or other data that would be too voluminous to describe are within the scope of this disclosure.

Other aspects or properties of digital pictures or pixels can be taken into account by Similarity Comparison **125** in digital picture comparisons. Examples of such aspects or properties include color adjustment, size adjustment, content manipulation, transparency (i.e. alpha channel, etc.), use of mask, and/or others. In some implementations, as digital pictures can be captured by various picture taking equipment, in various environments, and under various lighting conditions, Similarity Comparison **125** can adjust lighting or color of pixels or otherwise manipulate pixels before or during comparison. Lighting or color adjustment (also referred to as gray balance, neutral balance, white balance, etc.) may generally include manipulating or rebalancing the intensities of the colors (i.e. red, green, and/or blue if RGB color model is used, etc.) of one or more pixels. For example, Similarity Comparison **125** can adjust lighting or color of all pixels of one picture to make it more comparable to another picture. Similarity Comparison **125** can also incrementally adjust the pixels such as increasing or decreasing the red, green, and/or blue pixel values by a certain amount in each cycle of comparisons in order to find a substantially similar match at one of the incremental adjustment levels. Any of the publically available, custom, or other lighting or color adjustment techniques or programs can be utilized such as color filters, color balancing, color correction, and/or others. In other implementations, Similarity Comparison **125** can resize or otherwise transform a digital picture before or during comparison. Such resizing or transformation may include increasing or decreasing the number of pixels of a digital picture. For example, Similarity Comparison **125** can increase or decrease the size of a digital picture proportionally (i.e. increase or decrease length and/or width keeping aspect ratio constant, etc.) to equate its size with the size of another digital picture. Similarity Comparison **125** can also incrementally resize a digital picture such as increasing or decreasing the size of the digital picture proportionally by a certain amount in each cycle of comparisons in order to find a substantially similar match at one of the incremental sizes. Any of the publically available, custom, or other digital picture resizing techniques or programs can be utilized such as nearest-neighbor interpolation, bilinear interpolation, bicubic interpolation, and/or others. In further implementations, Similarity Comparison **125** can manipulate content (i.e. all pixels, one or more regions, one or more depicted objects/persons, etc.) of a digital picture before or during comparison. Such content manipulation may include moving, centering, aligning, resizing, transforming, and/or otherwise manipulating content of a digital picture. For example, Similarity Comparison **125** can move, center, or align content of one picture to make it more comparable to another picture. Any of the publically available, custom, or other digital picture manipulation techniques or programs can be utilized such as pixel moving, warping, distorting, aforementioned interpolations, and/or others. In further implementations, in digital pictures comprising transparency features or functionalities, Similarity Comparison **125** can utilize a threshold for acceptable number or percentage transparency difference similar to the below-described threshold for the acceptable color difference. Alternatively, transparency can be applied to one or more pixels of a digital picture and color difference may then be determined between compared pixels taking into account the transparency related color effect. Alternatively, transparent pixels can be excluded from comparison. In further implementations, certain regions or subsets of pixels can be ignored or excluded during comparison using a mask. In general, any region or subset of a picture determined to contain no content of interest can be excluded from comparison using a mask. Examples of such regions or subsets include background, transparent or partially transparent regions, regions comprising insignificant content, or any arbitrary region or subset. Similarity Comparison **125** can perform any other pre-processing or manipulation of digital pictures or pixels before or during comparison.

In any of the comparisons involving digital pictures or pixels, Similarity Comparison **125** can utilize a threshold for acceptable number or percentage difference in determining a match for each compared pixel. A pixel in a digital picture can be encoded using various techniques such as RGB (i.e. red, green, blue), CMYK (i.e. cyan, magenta, yellow, and key [black]), binary value, hexadecimal value, numeric value, and/or others. For instance, in RGB color scheme, each of red, green, and blue colors is encoded with a value 0-255 or its binary equivalent. In one example, a threshold for acceptable difference (i.e. absolute difference, etc.) can be set at 10 for each of the three colors. Therefore, a pixel encoded as R130, G240, B50 matches or is sufficiently similar to a compared pixel encoded as R135, G231, B57 because the differences in all three colors fall within the acceptable difference threshold (i.e. 10 in this example, etc.). Furthermore, a pixel encoded as R130, G240, B50 does not match or is not sufficiently similar to a compared pixel encoded as R143, G231, B57 because the difference in red value falls outside the acceptable difference threshold. Any other number threshold can be used such as 1, 3, 8, 15, 23, 77, 132, 197, 243, and/or others. A threshold for acceptable percentage difference can similarly be utilized such as 0.12%, 2%, 7%, 14%, 23%, 36%, 65%, and/or others. In some aspects, a threshold for acceptable number or percentage difference in red, green, and blue can be set to be different for each color. A similar difference determination can be utilized in pixels encoded in any other color scheme. The aforementioned thresholds can be defined by a user, by VSADO system administrator, or automatically by the sys-

tem based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input.

In some embodiments, Similarity Comparison **125** can compare one or more Extra Info **527** (i.e. time information, location information, computed information, observed information, sensory information, contextual information, and/or other information, etc.) in addition to or instead of comparing Digital Pictures **525** or portions thereof in determining substantial similarity of Knowledge Cells **800**. Extra Info **527** can be set to be less, equally, or more important (i.e. as indicated by importance index [later described], etc.) than Digital Pictures **525**, regions, features, pixels, and/or other elements in the comparison. Since Extra Info **527** may include any contextual or other information that can be useful in determining similarity of any compared elements, Extra Info **527** can be used to enhance any of the aforementioned similarity determinations.

In some embodiments, Similarity Comparison **125** can also compare one or more Instruction Sets **526** in addition to or instead of comparing Digital Pictures **525** or portions thereof in determining substantial similarity of Knowledge Cells **800**. In some aspects, Similarity Comparison **125** can compare portions of Instruction Sets **526** to determine substantial similarity of Instruction Sets **526**. Similar thresholds for the number or percentage of matching portions of the compared Instruction Sets **526** can be utilized in Instruction Set **526** comparisons. Such thresholds can be defined by a user, by VSADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In other aspects, Similarity Comparison **125** can compare text (i.e. character comparison, word/phrase search/comparison, semantic comparison, etc.) or other data (i.e. bit comparison, object or data structure comparison, etc.) to determine substantial similarity of Instruction Sets **526**. Any other comparison technique can be utilized in comparing Instruction Sets **526** in alternate implementations. Instruction Sets **526** can be set to be less, equally, or more important (i.e. as indicated by importance index [later described], etc.) than Digital Pictures **525**, regions, features, pixels, Extra Info **527**, and/or other elements in the comparison.

In some embodiments, an importance index (not shown) or other importance ranking technique can be used in any of the previously described comparisons or other processing involving elements of different importance. Importance index indicates importance of the element to or with which the index is assigned or associated. For example, importance index may indicate importance of a Knowledge Cell **800**, Digital Picture **525**, Instruction Set **526**, Extra Info **527**, region, feature, and/or other element to or with which the index is assigned or associated. In some aspects, importance index on a scale from 0 to 1 can be utilized, although, any other range can also be utilized. Importance index can be stored in or associated with the element to which the index pertains. Association of importance indexes can be implemented using a table where one column comprises elements and another column comprises their associated importance indexes, for example. Importance indexes of various elements can be defined by a user, by VSADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. In one example, a higher Importance index can be assigned to more substantive Digital Pictures **525** (i.e. pictures comprising content of interest [i.e. persons, objects, etc.], etc.). In another example, a higher importance index can be assigned to Digital Pictures **525** that are correlated with Instruction Sets **526**. Any importance index can be assigned to or associated with any element described herein. Any importance ranking technique can be utilized as or instead of importance index in alternate embodiments.

In some embodiments, Similarity Comparison **125** may generate a similarity index (not shown) for any compared elements. Similarity index indicates how well an element is matched with another element. For example, similarity index indicates how well a Knowledge Cell **800**, Digital Picture **527**, Instruction Set **526**, Extra Info **527**, region, feature, and/or other element is matched with a compared element. In some aspects, similarity index on a scale from 0 to 1 can be utilized, although, any other range can also be utilized. Similarity index can be generated by Similarity Comparison **125** whether substantial or other similarity between the compared elements is achieved or not. In one example, similarity index can be determined for a Knowledge Cell **800** based on a ratio/percentage of matched or substantially matched Digital Pictures **525** relative to the number of Digital Pictures **525** in the compared Knowledge Cell **800**. Specifically, similarity index of 0.93 is determined if 93% of Digital Pictures **525** of one Knowledge Cell **800** match or substantially match Digital Pictures **525** of another Knowledge Cell **800**. In some designs, importance (i.e. as indicated by importance index, etc.) of one or more Digital Pictures **525** can be included in the calculation of a weighted similarity index. Similar determination of similarity index can be implemented with Digital Pictures **525**, Instruction Sets **526**, Extra Info **527**, regions, features, pixels, and/or other elements or portions thereof. Any combination of the aforementioned similarity index determinations or calculations can be utilized in alternate embodiments. Any similarity ranking technique can be utilized to determine or calculate similarity index in alternate embodiments.

Referring to FIG. **20**, an embodiment of learning Knowledge Cells **800** comprising one or more Digital Pictures **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Neural Network **530*a*** comprising shortcut Connections **853** is illustrated. In some designs, Knowledge Cells **800** in one Layer **854** of Neural Network **530*a*** can be connected with Knowledge Cells **800** in any Layer **854**, not only in a successive Layer **854**, thereby creating shortcuts (i.e. shortcut Connections **853**, etc.) through Neural Network **530*a***. In some aspects, creating a shortcut Connection **853** can be implemented by performing Similarity Comparisons **125** of a Knowledge Cell **800** from Knowledge Structuring Unit **520** with Knowledge Cells **800** in any Layer **854** when applying (i.e. storing, copying, etc.) the Knowledge Cell **800** from Knowledge Structuring Unit **520** onto Neural Network **530*a***. Once created, shortcut Connections **853** enable a wider variety of Knowledge Cells **800** to be considered when selecting a path through Neural Network **530*a***. In some embodiments, Knowledge Structuring Unit **520** structures or generates Knowledge Cells **800** and the system applies them onto Neural Network **530*a***, thereby implementing learning Device's **98** operation in various visual surroundings. The system can perform Similarity Comparisons **125** of a Knowledge Cell **800** from Knowledge Structuring Unit **520** with Knowledge Cells **800** in a corresponding and/or other Layers **854** of Neural Network **530*a***. If a substantially similar Knowledge Cell **800** is not found in the corresponding or other Layers **854** of Neural Network **530*a***, the system may insert (i.e. copy, store, etc.) the Knowledge Cell **800** from Knowledge Structuring Unit **520** into the corresponding (or another) Layer **854** of Neural Network **530*a***, and create a Connection **853** to the inserted Knowledge Cell **800** from a prior Knowledge Cell **800** including assigning an occurrence count to the new Connection **853**,

calculating a weight of the new Connection **853**, and updating any other Connections **853** originating from the prior Knowledge Cell **800**. On the other hand, if a substantially similar Knowledge Cell **800** is found in the corresponding or other Layers **854** of Neural Network **530***a*, the system may update occurrence count and weight of Connection **853** to that Knowledge Cell **800** from a prior Knowledge Cell **800**, and update any other Connections **853** originating from the prior Knowledge Cell **800**. Any of the previously described and/or other techniques for comparing, inserting, updating, and/or other operations on Knowledge Cells **800**, Connections **853**, Layers **854**, and/or other elements can similarly be utilized in Neural Network **530***a* that comprises shortcut Connections **853**.

Referring to FIG. **21**, an embodiment of learning Knowledge Cells **800** comprising one or more Digital Pictures **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Graph **530***b* is illustrated. In some aspects, any Knowledge Cell **800** can be connected with any other Knowledge Cell **800** in Graph **530***b*. In other aspects, any Knowledge Cell **800** can be connected with itself and/or any other Knowledge Cell **800** in Graph **530***b*. In some embodiments, Knowledge Structuring Unit **520** structures or generates Knowledge Cells **800** and the system applies (i.e. store, copy, etc.) them onto Graph **530***b*, thereby implementing learning Device's **98** operation in various visual surroundings. The system can perform Similarity Comparisons **125** of a Knowledge Cell **800** from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Graph **530***b*. If a substantially similar Knowledge Cell **800** is not found in Graph **530***b*, the system may insert (i.e. copy, store, etc.) the Knowledge Cell **800** from Knowledge Structuring Unit **520** into Graph **530***b*, and create a Connection **853** to the inserted Knowledge Cell **800** from a prior Knowledge Cell **800** including assigning an occurrence count to the new Connection **853**, calculating a weight of the new Connection **853**, and updating any other Connections **853** originating from the prior Knowledge Cell **800**. On the other hand, if a substantially similar Knowledge Cell **800** is found in Graph **530***b*, the system may update occurrence count and weight of Connection **853** to that Knowledge Cell **800** from a prior Knowledge Cell **800**, and update any other Connections **853** originating from the prior Knowledge Cell **800**. Any of the previously described and/or other techniques for comparing, inserting, updating, and/or other operations on Knowledge Cells **800**, Connections **853**, and/or other elements can similarly be utilized in Graph **530***b*.

For example, the system can perform Similarity Comparisons **125** of Knowledge Cell **800***ba* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Graph **530***b*. In the case that a substantially similar match is not found, the system may insert Knowledge Cell **800***ha* into Graph **530***b* and copy Knowledge Cell **800***ba* into the inserted Knowledge Cell **800***ha*. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bb* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Graph **530***b*. In the case that a substantially similar match is found between Knowledge Cell **800***bb* and Knowledge Cell **800***hb*, the system may create Connection **853***h***1** between Knowledge Cell **800***ha* and Knowledge Cell **800***hb* with occurrence count of 1 and weight of 1. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bc* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Graph **530***b*. In the case that a substantially similar match is found between Knowledge Cell **800***bc* and Knowledge Cell **800***hc*, the system may update occurrence count and weight of Connection **853***h***2** between Knowledge Cell **800***hb* and Knowledge Cell **800***hc*, and update weights of other outgoing Connections **853** (one in this example) originating from Knowledge Cell **800***hb* as previously described. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***bd* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Graph **530***b*. In the case that a substantially similar match is not found, the system may insert Knowledge Cell **800***hd* into Graph **530***b* and copy Knowledge Cell **800***bd* into the inserted Knowledge Cell **800***hd*. The system may also create Connection **853***h***3** between Knowledge Cell **800***hc* and Knowledge Cell **800***hd* with occurrence count of 1 and weight calculated based on the occurrence count as previously described. The system may also update weights of other outgoing Connections **853** (one in this example) originating from Knowledge Cell **800***hc* as previously described. The system can then perform Similarity Comparisons **125** of Knowledge Cell **800***be* from Knowledge Structuring Unit **520** with Knowledge Cells **800** in Graph **530***b*. In the case that a substantially similar match is not found, the system may insert Knowledge Cell **800***he* into Graph **530***b* and copy Knowledge Cell **800***be* into the inserted Knowledge Cell **800***he*. The system may also create Connection **853***h***4** between Knowledge Cell **800***hd* and Knowledge Cell **800***he* with occurrence count of 1 and weight of 1. Applying any additional Knowledge Cells **800** from Knowledge Structuring Unit **520** onto Graph **530***b* follows similar logic or process as the above-described.

Referring to FIG. **22**, an embodiment of learning Knowledge Cells **800** comprising one or more Digital Pictures **525** correlated with any Instruction Sets **526** and/or Extra Info **527** using Collection of Sequences **530***c* is illustrated. Collection of Sequences **530***c* comprises the functionality for storing one or more Sequences **533**. Sequence **533** comprises the functionality for storing multiple Knowledge Cells **800**. In some aspects, a Sequence **533** may include Knowledge Cells **800** relating to a single operation of Device **98**. For example, Knowledge Structuring Unit **520** structures or generates Knowledge Cells **800** and the system applies them onto Collection of Sequences **530***c*, thereby implementing learning Device's **98** operation in various visual surroundings. The system can perform Similarity Comparisons **125** of Knowledge Cells **800** from Knowledge Structuring Unit **520** with corresponding Knowledge Cells **800** in Sequences **533** of Collection of Sequences **530***c* to find a Sequence **533** comprising Knowledge Cells **800** that are substantially similar to the Knowledge Cells **800** from Knowledge Structuring Unit **520**. If Sequence **533** comprising such substantially similar Knowledge Cells **800** is not found in Collection of Sequences **530***c*, the system may create a new Sequence **533** comprising the Knowledge Cells **800** from Knowledge Structuring Unit **520** and insert (i.e. copy, store, etc.) the new Sequence **533** into Collection of Sequences **530***c*. On the other hand, if Sequence **533** comprising substantially similar Knowledge Cells **800** is found in Collection of Sequences **530***c*, the system may optionally omit inserting the Knowledge Cells **800** from Knowledge Structuring Unit **520** into Collection of Sequences **530***c* as inserting a similar Sequence **533** may not add much or any additional knowledge. This approach can save storage resources and limit the number of Knowledge Cells **800** that may later need to be processed or compared. In other aspects, a Sequence **533** may include Knowledge Cells **800** relating to a part of an operation of Device **98**. Similar learning process as the above described can be utilized in such implementations. In further aspects, one or more long Sequences **533** each including Knowledge Cells **800** of

multiple operations of Device **98** can be utilized. In one example, Knowledge Cells **800** of all operations can be stored in a single long Sequence **533** in which case Collection of Sequences **530***c* as a separate element can be omitted. In another example, Knowledge Cells **800** of multiple operations can be included in a plurality of long Sequences **533** such as hourly, daily, weekly, monthly, yearly, or other periodic or other Sequences **533**. Similarity Comparisons **125** can be performed by traversing the one or more long Sequences **533** to find a match or substantially similar match. For instance, the system can perform Similarity Comparisons **125** of Knowledge Cells **800** from Knowledge Structuring Unit **520** with corresponding Knowledge Cells **800** in subsequences of a long Sequence **533** in incremental or other traversing pattern to find a subsequence comprising Knowledge Cells **800** that are substantially similar to the Knowledge Cells **800** from Knowledge Structuring Unit **520**. The incremental traversing pattern may start from one end of a long Sequence **533** and move the comparison subsequence up or down one or any number of incremental Knowledge Cells **800** at a time. Other traversing patterns or methods can be employed such as starting from the middle of the Sequence **533** and subdividing the resulting subsequences in a recursive pattern, or any other traversing pattern or method. If a subsequence comprising substantially similar Knowledge Cells **800** is not found in the long Sequence **533**, the system may concatenate or append the Knowledge Cells **800** from Knowledge Structuring Unit **520** to the long Sequence **533**. In further aspects, Connections **853** can optionally be used in Sequence **533** to connect Knowledge Cells **800**. For example, a Knowledge Cell **800** can be connected not only with a next Knowledge Cell **800** in the Sequence **533**, but also with any other Knowledge Cell **800** in the Sequence **533**, thereby creating alternate routes or shortcuts through the Sequence **533**. Any number of Connections **853** connecting any Knowledge Cells **800** can be utilized. Any of the previously described and/or other techniques for comparing, inserting, updating, and/or other operations on Knowledge Cells **800**, Connections **853**, and/or other elements can similarly be utilized in Sequences **533** and/or Collection of Sequences **530***c*.

Any of the previously described data structures or arrangements of Knowledge Cells **800** such as Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, and/or others can be used alone, or in combination with each other or with other elements, in alternate embodiments. In one example, a path in Neural Network **530***a* or Graph **530***b* may include its own separate sequence of Knowledge Cells **800** that are not interconnected with Knowledge Cells **800** in other paths. In another example, a part of a path in Neural Network **530***a* or Graph **530***b* may include a sequence of Knowledge Cells **800** interconnected with Knowledge Cells **800** in other paths, whereas, another part of the path may include its own separate sequence of Knowledge Cells **800** that are not interconnected with Knowledge Cells **800** in other paths. Any other combinations or arrangements of Knowledge Cells **800** can be implemented.

Referring to FIG. **23**, an embodiment of determining anticipatory Instruction Sets **526** from a single Knowledge Cell **800** is illustrated. Knowledge Cell **800** may be part of a Knowledgebase **530** (i.e. Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.) such as Collection of Knowledge Cells **530***d*. Decision-making Unit **540** comprises the functionality for anticipating or determining a device's operation in various visual surroundings. Decision-making Unit **540** comprises the functionality for anticipating or determining Instruction Sets **526** (i.e. anticipatory Instruction Sets **526**, etc.) to be used or executed in Device's **98** autonomous operation based on incoming Digital Pictures **525** of Device's **98** visual surrounding. Decision-making Unit **540** also comprises other disclosed functionalities.

In some aspects, Decision-making Unit **540** may anticipate or determine Instruction Sets **526** (i.e. anticipatory Instruction Sets **526**, etc.) for autonomous Device **98** operation by performing Similarity Comparisons **125** of incoming Digital Pictures **525** or portions thereof from Picture Capturing Apparatus **90** with Digital Pictures **525** or portions thereof from Knowledge Cells **800** in Knowledgebase **530** (i.e. Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.). A Knowledge Cell **800** includes a unit of knowledge (i.e. one or more Digital Pictures **525** correlated with any Instruction Sets **526** and/or Extra Info **527**, etc.) of how Device **98** operated in a visual surrounding as previously described. When Digital Pictures **525** or portions thereof of a similar visual surrounding are detected in the future, Decision-making Unit **540** can anticipate the Instruction Sets **526** (i.e. anticipatory Instruction Sets **526**, etc.) previously learned in a similar visual surrounding, thereby enabling autonomous Device **98** operation. In some aspects, Decision-making Unit **540** can perform Similarity Comparisons **125** of incoming Digital Pictures **525** from Picture Capturing Apparatus **90** with Digital Pictures **525** from Knowledge Cells **800** in Knowledgebase **530** (i.e. Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.). If one or more substantially similar Digital Pictures **525** or portions thereof are found in a Knowledge Cell **800** from Knowledgebase **530**, Instruction Sets **526** (i.e. anticipatory Instruction Sets **526**, etc.) for autonomous Device **98** operation can be anticipated in Instruction Sets **526** correlated with the one or more Digital Pictures **525** from the Knowledge Cell **800**. In some designs, subsequent one or more Instruction Sets **526** for autonomous Device **98** operation can be anticipated in Instruction Sets **526** correlated with subsequent Digital Pictures **525** from the Knowledge Cell **800** (or other Knowledge Cells **800**), thereby anticipating not only current, but also additional future Instruction Sets **526**. Although, Extra Info **527** is not shown in this and/or other figures for clarity of illustration, it should be noted that any Digital Picture **525**, Instruction Set **526**, and/or other element may include or be associated with Extra Info **527** and that Decision-making Unit **540** can utilize Extra Info **527** for enhanced decision making.

For example, Decision-making Unit **540** can perform Similarity Comparisons **125** of Digital Picture **52511** or portion thereof from Picture Capturing Apparatus **90** with Digital Picture **525***a***1** or portion thereof from Knowledge Cell **800***oa*. Digital Picture **525***a***1** or portion thereof from Knowledge Cell **800***oa* may be found substantially similar. Decision-making Unit **540** can anticipate Instruction Sets **526***a***1**-**526***a***3** correlated with Digital Picture **525***a***1**, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Digital Picture **52512** or portion thereof from Picture Capturing Apparatus **90** with Digital Picture **525***a***2** or portion thereof from Knowledge Cell **800***oa*. Digital Picture **525***a***2** or portion thereof from Knowledge Cell **800***oa* may be found substantially similar. Decision-making Unit **540** can anticipate Instruction Set **526***a***4** correlated with Digital Picture **525***a***2**, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform

Similarity Comparisons **125** of Digital Picture **52513** or portion thereof from Picture Capturing Apparatus **90** with Digital Picture **525*a*3** or portion thereof from Knowledge Cell **800*oa***. Digital Picture **525*a*3** or portion thereof from Knowledge Cell **800*oa*** may be found substantially similar. Decision-making Unit **540** may not anticipate any Instruction Sets **526** since none are correlated with Digital Picture **525*a*3**. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Digital Picture **52514** or portion thereof from Picture Capturing Apparatus **90** with Digital Picture **525*a*4** or portion thereof from Knowledge Cell **800*oa***. Digital Picture **525*a*4** or portion thereof from Knowledge Cell **800*oa*** may not be found substantially similar. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Digital Picture **52515** or portion thereof from Picture Capturing Apparatus **90** with Digital Picture **525*a*5** or portion thereof from Knowledge Cell **800*oa***. Digital Picture **525*a*5** or portion thereof from Knowledge Cell **800*oa*** may not be found substantially similar. Decision-making Unit **540** can implement similar logic or process for any additional Digital Picture **525** from Picture Capturing Apparatus **90**, and so on.

It should be understood that any of the described elements and/or techniques in the foregoing example can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells **800** or elements (i.e. Digital Pictures **525**, Instruction Sets **526**, etc.) thereof would be affected accordingly. In one example, Extra Info **527** can be included in the Similarity Comparisons **125** as previously described. In another example, as history of incoming Digital Pictures **525** becomes available, Decision-making Unit **540** can perform collective Similarity Comparisons **125** of the history of Digital Pictures **525** or portions thereof from Picture Capturing Apparatus **90** with subsequences of Digital Pictures **525** or portions thereof from Knowledge Cell **800**. In a further example, the described comparisons in a single Knowledge Cell **800** may be performed on any number of Knowledge Cells **800** sequentially or in parallel. Parallel processors such as a plurality of Processors **11** or cores thereof can be utilized for such parallel processing. In a further example, various arrangements of Digital Pictures **525** and/or other elements in a Knowledge Cell **800** can be utilized as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to FIG. **24**, an embodiment of determining anticipatory Instruction Sets **526** by traversing a single Knowledge Cell **800** is illustrated. Knowledge Cell **800** may be part of a Knowledgebase **530** (i.e. Neural Network **530*a***, Graph **530*b***, Collection of Sequences **530*c***, Sequence **533**, Collection of Knowledge Cells **530*d***, etc.) such as Collection of Knowledge Cells **530*d***. For example, Decision-making Unit **540** can perform Similarity Comparisons **125** of Digital Picture **52511** or portion thereof from Picture Capturing Apparatus **90** with Digital Picture **525*a*1** or portion thereof from Knowledge Cell **800*oa***. Digital Picture **525*a*1** or portion thereof from Knowledge Cell **800*oa*** may not be found substantially similar. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Digital Picture **52511** or portion thereof from Picture Capturing Apparatus **90** with Digital Picture **525*a*2** or portion thereof from Knowledge Cell **800*oa***. Digital Picture **525*a*2** or portion thereof from Knowledge Cell **800*oa*** may not be found substantially similar. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Digital Picture **52511** or portion thereof from Picture Capturing Apparatus **90** with Digital Picture **525*a*3** or portion thereof from Knowledge Cell **800*oa***. Digital Picture **525*a*3** or portion thereof from Knowledge Cell **800*oa*** may be found substantially similar. Decision-making Unit **540** may not anticipate any Instruction Sets **526** since none are correlated with Digital Picture **525*a*3**. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Digital Picture **52512** or portion thereof from Picture Capturing Apparatus **90** with Digital Picture **525*a*4** or portion thereof from Knowledge Cell **800*oa***. Digital Picture **525*a*4** or portion thereof from Knowledge Cell **800*oa*** may be found substantially similar. Decision-making Unit **540** can anticipate Instruction Sets **526*a*5-526*a*6** correlated with Digital Picture **525*a*4**, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Digital Picture **52513** or portion thereof from Picture Capturing Apparatus **90** with Digital Picture **525*a*5** or portion thereof from Knowledge Cell **800*oa***. Digital Picture **525*a*5** or portion thereof from Knowledge Cell **800*oa*** may be found substantially similar. Decision-making Unit **540** may not anticipate any Instruction Sets **526** since none are correlated with Digital Picture **525*a*5**. Decision-making Unit **540** can implement similar logic or process for any additional Digital Pictures **525** from Picture Capturing Apparatus **90**, and so on.

It should be understood that any of the described elements and/or techniques in the foregoing example can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells **800** or elements (i.e. Digital Pictures **525**, Instruction Sets **526**, etc.) thereof would be affected accordingly. In one example, Extra Info **527** can be included in the Similarity Comparisons **125** as previously described. In another example, as history of incoming Digital Pictures **525** becomes available, Decision-making Unit **540** can perform collective Similarity Comparisons **125** of the history of Digital Pictures **525** or portions thereof from Picture Capturing Apparatus **90** with subsequences of Digital Pictures **525** or portions thereof from Knowledge Cell **800**. In a further example, traversing may be performed in incremental traversing pattern such as starting from one end of Knowledge Cell **800** and moving the comparison subsequence up or down the list one or any number of incremental Digital Pictures **525** at a time. Other traversing patterns or methods can be employed such as starting from the middle of the Knowledge Cell **800** and subdividing the resulting subsequence in a recursive pattern, or any other traversing pattern or method. In a further example, the described traversing of a single Knowledge Cell **800** may be performed on any number of Knowledge Cells **800** sequentially or in parallel. Parallel processors such as a plurality of Processors **11** or cores thereof can be utilized for such parallel processing. In a further example, various arrangements of Digital Pictures **525** and/or other elements in a Knowledge Cell **800** can be utilized as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to FIG. **25**, an embodiment of determining anticipatory Instruction Sets **526** using collective similarity comparisons is illustrated. For example, Decision-making Unit **540** can perform Similarity Comparisons **125** of Digital

Picture **52511** or portion thereof from Picture Capturing Apparatus **90** with corresponding Digital Pictures **525** or portions thereof from Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. Digital Picture **525***c***1** or portion thereof from Knowledge Cell **800***rc* may be found substantially similar with highest similarity. Decision-making Unit **540** can anticipate any Instruction Sets **526** (not shown) correlated with Digital Picture **525***c***1**, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform collective Similarity Comparisons **125** of Digital Pictures **52511**-**52512** or portions thereof from Picture Capturing Apparatus **90** with corresponding Digital Pictures **525** or portions thereof from Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. Digital Pictures **525***c***1**-**525***c***2** or portions thereof from Knowledge Cell **800***rc* may be found substantially similar with highest similarity. Decision-making Unit **540** can anticipate any Instruction Sets **526** (not shown) correlated with Digital Picture **525***c***2**, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform collective Similarity Comparisons **125** of Digital Pictures **52511**-**52513** or portions thereof from Picture Capturing Apparatus **90** with corresponding Digital Pictures **525** or portions thereof from Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. Digital Pictures **525***d***1**-**525***d***3** or portions thereof from Knowledge Cell **800***rd* may be found substantially similar with highest similarity. Decision-making Unit **540** can anticipate any Instruction Sets **526** (not shown) correlated with Digital Picture **525***d***3**, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform collective Similarity Comparisons **125** of Digital Pictures **52511**-**52514** or portions thereof from Picture Capturing Apparatus **90** with corresponding Digital Pictures **525** or portions thereof from Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. Digital Pictures **525***d***1**-**525***d***4** or portions thereof from Knowledge Cell **800***rd* may be found substantially similar with highest similarity. Decision-making Unit **540** can anticipate any Instruction Sets **526** (not shown) correlated with Digital Picture **525***d***4**, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform collective Similarity Comparisons **125** of Digital Pictures **52511**-**52515** or portions thereof from Picture Capturing Apparatus **90** with corresponding Digital Pictures **525** or portions thereof from Knowledge Cells **800** in Collection of Knowledge Cells **530***d*. Digital Pictures **525***d***1**-**525***d***5** or portions thereof from Knowledge Cell **800***rd* may be found substantially similar with highest similarity. Decision-making Unit **540** can anticipate any Instruction Sets **526** (not shown) correlated with Digital Picture **525***d***5**, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can implement similar logic or process for any additional Digital Picture **525** from Picture Capturing Apparatus **90**, and so on.

In some embodiments, various elements and/or techniques can be utilized in the aforementioned similarity determinations with respect to collectively compared Digital Pictures **525** and/or other elements. In some aspects, similarity of collectively compared Digital Pictures **525** can be determined based on similarities or similarity indexes of the individually compared Digital Pictures **525**. In one example, an average of similarities or similarity indexes of individually compared Digital Pictures **525** can be used to determine similarity of collectively compared Digital Pictures **525**. In another example, a weighted average of similarities or similarity indexes of individually compared Digital Pictures **525** can be used to determine similarity of collectively compared Digital Pictures **525**. For instance, to affect the weighting of collective similarity, a higher weight or importance (i.e. importance index, etc.) can be assigned to the similarities or similarity indexes of some (i.e. more substantive, etc.) Digital Pictures **525** and lower for other (i.e. less substantive, etc.) Digital Pictures **525**. Any other higher or lower weight or importance assignment can be implemented. In other aspects, any of the previously described or other thresholds for substantial similarity of individually compared elements can be similarly utilized for collectively compared elements. In one example, substantial similarity of collectively compared Digital Pictures **525** can be achieved when their collective similarity or similarity index exceeds a similarity threshold. In another example, substantial similarity of collectively compared Digital Pictures **525** can be achieved when at least a threshold number or percentage of Digital Pictures **525** or portions thereof of the collectively compared Digital Pictures **525** match or substantially match. Similarly, substantial similarity of collectively compared Digital Pictures **525** can be achieved when a number or percentage of matching or substantially matching Digital Pictures **525** or portions thereof of the collectively compared Digital Pictures **525** exceeds a threshold. Such thresholds can be defined by a user, by VSADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. Similar elements and/or techniques as the aforementioned can be used for similarity determinations of other collectively compared elements such as Instruction Sets **526**, Extra Info **527**, Knowledge Cells **800**, and/or others. Similarity determinations of collectively compared elements may include any features, functionalities, and embodiments of Similarity Comparison **125**, and vice versa.

It should be understood that any of the described elements and/or techniques in the foregoing example can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells **800** or elements (i.e. Digital Pictures **525**, Instruction Sets **526**, etc.) thereof would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info **527** in Similarity Comparisons **125**, traversing of Knowledge Cells **800** or other elements, using history of Digital Pictures **525** or Knowledge Cells **800** for collective Similarity Comparisons **125**, using various arrangements of Digital Pictures **525** and/or other elements in a Knowledge Cell **800**, and/or others can similarly be utilized in this example. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to FIG. **26**, an embodiment of determining anticipatory Instruction Sets **526** using Neural Network **530***a* is illustrated. In some aspects, determining anticipatory Instruction Sets **526** using Neural Network **530***a* may include selecting a path of Knowledge Cells **800** or elements (i.e. Digital Pictures **525**, Instruction Sets **526**, etc.) thereof through Neural Network **530***a*. Decision-making Unit **540** can utilize various elements and/or techniques for selecting a path through Neural Network **530***a*. Although, these elements and/or techniques are described using Neural Network **530***a* below, they can similarly be used in any Knowledgebase **530** (i.e. Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.) where applicable.

In some embodiments, Decision-making Unit **540** can utilize similarity index in selecting Knowledge Cells **800** or elements (i.e. Digital Pictures **525**, Instruction Sets **526**, etc.) thereof in a path through Neural Network **530***a*. For instance, similarity index may indicate how well one or more Digital Pictures **525** or portions thereof are matched with one or more other Digital Pictures **525** or portions thereof as previously described. In one example, Decision-making Unit **540** may select a Knowledge Cell **800** comprising one or more Digital Pictures **525** with highest similarity index even if Connection **853** pointing to that Knowledge Cell **800** has less than the highest weight. Therefore, similarity index or other such element or parameter can override or disregard the weight of a Connection **853** or other element. In another example, Decision-making Unit **540** may select a Knowledge Cell **800** comprising one or more Digital Pictures **525** whose similarity index is higher than or equal to a weight of Connection **853** pointing to that Knowledge Cell **800**. In a further example, Decision-making Unit **540** may select a Knowledge Cell **800** comprising one or more Digital Pictures **525** whose similarity index is lower than or equal to a weight of Connection **853** pointing to that Knowledge Cell **800**. Similarity index can be set to be more, less, or equally important than a weight of a Connection **853**.

In other embodiments, Decision-making Unit **540** can utilize Connections **853** in selecting Knowledge Cells **800** or elements (i.e. Digital Pictures **525**, Instruction Sets **526**, etc.) thereof in a path through Neural Network **530***a*. In some aspects, Decision-making Unit **540** can take into account weights of Connections **853** among the interconnected Knowledge Cells **800** in choosing from which Knowledge Cell **800** to compare one or more Digital Pictures **525** first, second, third, and so on. Specifically, for instance, Decision-making Unit **540** can perform Similarity Comparison **125** with one or more Digital Pictures **525** from Knowledge Cell **800** pointed to by the highest weight Connection **853** first, Digital Pictures **525** from Knowledge Cell **800** pointed to by the second highest weight Connection **853** second, and so on. In other aspects, Decision-making Unit **540** can stop performing Similarity Comparisons **125** as soon as it finds one or more substantially similar Digital Pictures **525** in an interconnected Knowledge Cell **800**. In further aspects, Decision-making Unit **540** may only follow the highest weight Connection **853** to arrive at a Knowledge Cell **800** comprising one or more Digital Pictures **525** to be compared, thereby disregarding Connections **853** with less than the highest weight. In further aspects, Decision-making Unit **540** may ignore Connections **853** and/or their weights.

In further embodiments, Decision-making Unit **540** can utilize a bias to adjust similarity index, weight of a Connection **853**, and/or other element or parameter used in selecting Knowledge Cells **800** or elements (i.e. Digital Pictures **525**, Instruction Sets **526**, etc.) thereof in a path through Neural Network **530***a*. In one example, Decision-making Unit **540** may select a Knowledge Cell **800** comprising one or more Digital Pictures **525** whose similarity index multiplied by or adjusted for a bias is higher than or equal to a weight of Connection **853** pointing to that Knowledge Cell **800**. In another example, Decision-making Unit **540** may select a Knowledge Cell **800** comprising one or more Digital Pictures **525** whose similarity index multiplied by or adjusted for a bias is lower than or equal to a weight of Connection **853** pointing to that Knowledge Cell **800**. In a further example, bias can be used to resolve deadlock situations where similarity index is equal to a weight of a Connection **853**. In some aspects, bias can be expressed in percentages such as 0.3 percent, 1.2 percent, 25.7 percent, 79.8 percent, 99.9 percent, 100.1 percent, 155.4 percent, 298.6 percent, 1105.5 percent, and so on. For example, a bias below 100 percent decreases an element or parameter to which it is applied, a bias equal to 100 percent does not change the element or parameter to which it is applied, and a bias higher than 100 percent increases the element or parameter to which it is applied. In general, any amount of bias can be utilized. Bias can be applied to one or more of a weight of a Connection **853**, similarity index, any other element or parameter, and/or all or any combination of them. Also, different biases can be applied to each of a weight of a Connection **853**, similarity index, or any other element or parameter. For example, 30 percent bias can be applied to similarity index and 15 percent bias can be applied to a weight of a Connection **853**. Also, different biases can be applied to various Layers **854** of Neural Network **530***a*, and/or other disclosed elements. Bias can be defined by a user, by VSADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input.

Any other element and/or technique can be utilized in selecting Knowledge Cells **800** or elements (i.e. Digital Pictures **525**, Instruction Sets **526**, etc.) thereof in a path through Neural Network **530***a*.

In some embodiments, Neural Network **530***a* may include knowledge (i.e. interconnected Knowledge Cells **800** comprising one or more Digital Pictures **525** correlated with any Instruction Sets **526** and/or Extra Info **527**, etc.) of how Device **98** operated in various visual surroundings. In some aspects, determining anticipatory Instruction Sets **526** using Neural Network **530***a* may include selecting a path of Knowledge Cells **800** or elements (i.e. Digital Pictures **525**, Instruction Sets **526**, etc.) thereof through Neural Network **530***a*. Individual and/or collective Similarity Comparisons **125** can be used to determine substantial similarity of the individually and/or collectively compared Digital Pictures **525** or portions thereof. Substantial similarity may be used primarily for selecting a path through Neural Network **530***a*, whereas, weight of any Connection **853** may be used secondarily or not at all.

For example, Decision-making Unit **540** can perform Similarity Comparisons **125** of Digital Pictures **525***a***1**-**525***an* or portions thereof from Picture Capturing Apparatus **90** with Digital Pictures **525** or portions thereof from one or more Knowledge Cells **800** in Layer **854***a* (or any other one or more Layers **854**, etc.). Digital Pictures **525** or portions thereof from Knowledge Cell **800***ta* may be found collectively substantially similar with highest similarity. As the comparisons of individual Digital Pictures **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Digital Pictures **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Digital Pictures **525***b***1**-**525***bn* or portions thereof from Picture Capturing Apparatus **90** with Digital Pictures **525** or portions thereof from one or more Knowledge Cells **800** in Layer **854***b* interconnected with Knowledge Cell **800***ta*. Digital Pictures **525** or portions thereof from Knowledge Cell **800***tb* may be found collectively substantially similar with highest similarity, thus, Decision-making Unit **540** may follow Connection **853***t***1** disregarding its less than highest weight. As the comparisons of individual Digital Pictures **525** are performed to determine collective similarity, Decision-making Unit **540** can

anticipate Instruction Sets **526** correlated with substantially similar individual Digital Pictures **525** as previously described, thereby enabling autonomous Device **98** operation. Since Connection **853*t*2** is the only connection from Knowledge Cell **800*tb***, Decision-making Unit **540** may follow Connection **853*t*2** and perform Similarity Comparisons **125** of Digital Pictures **525*c*1**-**525*cn*** or portions thereof from Picture Capturing Apparatus **90** with Digital Pictures **525** or portions thereof from Knowledge Cell **800*tc*** in Layer **854*c***. Digital Pictures **525** or portions thereof from Knowledge Cell **800*tc*** may be found collectively substantially similar. As the comparisons of individual Digital Pictures **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Digital Pictures **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Digital Pictures **525*d*1**-**525*dn*** or portions thereof from Picture Capturing Apparatus **90** with Digital Pictures **525** or portions thereof from one or more Knowledge Cells **800** in Layer **854*d*** interconnected with Knowledge Cell **800*tc***. Digital Pictures **525** or portions thereof from Knowledge Cell **800*td*** may be found collectively substantially similar with highest similarity, thus, Decision-making Unit **540** may follow Connection **853*t*3**. As the comparisons of individual Digital Pictures **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Digital Pictures **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Digital Pictures **525*e*1**-**525*en*** or portions thereof from Picture Capturing Apparatus **90** with Digital Pictures **525** or portions thereof from one or more Knowledge Cells **800** in Layer **854*e*** interconnected with Knowledge Cell **800*td***. Digital Pictures **525** or portions thereof from Knowledge Cell **800*te*** may be found collectively substantially similar with highest similarity, thus, Decision-making Unit **540** may follow Connection **853*t*4**. As the comparisons of individual Digital Pictures **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Digital Pictures **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can implement similar logic or process for any additional Digital Pictures **525** from Picture Capturing Apparatus **90**, and so on.

The foregoing exemplary embodiment provides an example of utilizing a combination of collective Similarity Comparisons **125**, individual Similarity Comparisons **125**, Connections **853**, and/or other elements or techniques. It should be understood that any of these elements and/or techniques can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells **800** or elements (i.e. Digital Pictures **525**, Instruction Sets **526**, etc.) thereof in a path through Neural Network **530*a*** would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info **527** in Similarity Comparisons **125**, traversing of Knowledge Cells **800** or other elements, using history of Digital Pictures **525** or Knowledge Cells **800** for collective Similarity Comparisons **125**, using various arrangements of Digital Pictures **525** and/or other elements in a Knowledge Cell **800**, and/or others can similarly be utilized in this example. These elements and/or techniques can similarly be utilized in Graph **530*b***, Collection of Sequences **530*c***, Sequence **533**, Collection of Knowledge Cells **530*d***, and/or other data structures or arrangements. In some aspects, instead of anticipating Instruction Sets **526** correlated with substantially similar individual Digital Pictures **525**, Decision-making Unit **540** can anticipate instruction Sets **526** correlated with substantially similar streams of Digital Pictures **525**. In other aspects, any time that substantial similarity or other similarity threshold is not achieved in compared Digital Pictures **525** or portions thereof of any of the Knowledge Cells **800**, Decision-making Unit **540** can decide to look for a substantially or otherwise similar Digital Pictures **525** or portions thereof in Knowledge Cells **800** elsewhere in Neural Network **530*a*** such as in any Layer **854** subsequent to a current Layer **854**, in the first Layer **854**, in the entire Neural Network **530*a***, and/or others, even if such Knowledge Cell **800** may be unconnected with a prior Knowledge Cell **800**. It should be noted that any of Digital Pictures **525*a*1**-**525*an***, Digital Pictures **525*b*1**-**525*bn***, Digital Pictures **525*c*1**-**525*cn***, Digital Pictures **525*d*1**-**525*dn***, Digital Pictures **525*e*1**-**525*en***, etc. may include one Digital Picture **525** or a stream of Digital Pictures **525**. It should also be noted that any Knowledge Cell **800** may include one Digital Picture **525** or a stream of Digital Pictures **525** as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to FIG. **27**, an embodiment of determining anticipatory Instruction Sets **526** using Graph **530*b*** is illustrated. Graph **530*b*** may include knowledge (i.e. interconnected Knowledge Cells **800** comprising one or more Digital Pictures **525** correlated with any Instruction Sets **526** and/or Extra Info **527**, etc.) of how Device **98** operated in various visual surroundings. In some aspects, determining anticipatory Instruction Sets **526** using Graph **530*b*** may include selecting a path of Knowledge Cells **800** or elements (i.e. Digital Pictures **525**, Instruction Sets **526**, etc.) thereof through Graph **530*b***. Individual and/or collective Similarity Comparisons **125** can be used to determine substantial similarity of the individually and/or collectively compared Digital Pictures **525** or portions thereof. Substantial similarity may be used primarily for selecting a path through Graph **530*b***, whereas, weight of any Connection **853** may be used secondarily or not at all.

For example, Decision-making Unit **540** can perform Similarity Comparisons **125** of Digital Pictures **525*a*1**-**525*an*** or portions thereof from Picture Capturing Apparatus **90** with Digital Pictures **525** or portions thereof from one or more Knowledge Cells **800** in Graph **530*b***. Digital Pictures **525** or portions thereof from Knowledge Cell **800*ua*** may be found collectively substantially similar with highest similarity. As the comparisons of individual Digital Pictures **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Digital Pictures **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Digital Pictures **525*b*1**-**525*bn*** or portions thereof from Picture Capturing Apparatus **90** with Digital Pictures **525** or portions thereof from one or more Knowledge Cells **800** in Graph **530*b*** interconnected with Knowledge Cell **800*ua*** by outgoing Connections **853**. Digital Pictures **525** or portions thereof from Knowledge Cell **800*ub*** may be found collectively

substantially similar with highest similarity, thus, Decision-making Unit **540** may follow Connection **853***u***1** disregarding its less than highest weight. As the comparisons of individual Digital Pictures **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Digital Pictures **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Digital Pictures **525***c***1**-**525***cn* or portions thereof from Picture Capturing Apparatus **90** with Digital Pictures **525** or portions thereof from one or more Knowledge Cells **800** in Graph **530***b* interconnected with Knowledge Cell **800***ub* by outgoing Connections **853**. Digital Pictures **525** or portions thereof from Knowledge Cell **800***uc* may be found collectively substantially similar with highest similarity, thus, Decision-making Unit **540** may follow Connection **853***u***2** disregarding its less than highest weight. As the comparisons of individual Digital Pictures **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Digital Pictures **525** as previously described, thereby enabling autonomous Device **98** operation. Since Connection **853***u***3** is the only connection from Knowledge Cell **800***uc*, Decision-making Unit **540** may follow Connection **853***u***3** and perform Similarity Comparisons **125** of Digital Pictures **525***d***1**-**525***dn* or portions thereof from Picture Capturing Apparatus **90** with Digital Pictures **525** or portions thereof from Knowledge Cell **800***ud* in Graph **530***b*. Digital Pictures **525** or portions thereof from Knowledge Cell **800***ud* may be found collectively substantially similar. As the comparisons of individual Digital Pictures **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Digital Pictures **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Digital Pictures **525***e***1**-**525***en* or portions thereof from Picture Capturing Apparatus **90** with Digital Pictures **525** or portions thereof from one or more Knowledge Cells **800** in Graph **530***b* interconnected with Knowledge Cell **800***ud* by outgoing Connections **853**. Digital Pictures **525** or portions thereof from Knowledge Cell **800***ue* may be found collectively substantially similar with highest similarity, thus, Decision-making Unit **540** may follow Connection **853***u***4**. As the comparisons of individual Digital Pictures **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Digital Pictures **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can implement similar logic or process for any additional Digital Pictures **525** from Activity Detector **160**, and so on.

The foregoing exemplary embodiment provides an example of utilizing a combination of collective Similarity Comparisons **125**, individual Similarity Comparisons **125**, Connections **853**, and/or other elements or techniques. It should be understood that any of these elements and/or techniques can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells **800** or elements (i.e. Digital Pictures **525**, Instruction Sets **526**, etc.) thereof in a path through Graph **530***b* would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info **527** in Similarity Comparisons **125**, traversing of Knowledge Cells **800** or other elements, using history of Digital Pictures **525** or Knowledge Cells **800** in collective Similarity Comparisons **125**, using various arrangements of Digital Pictures **525** and/or other elements in a Knowledge Cell **800**, and/or others can similarly be utilized in this example. These elements and/or techniques can similarly be utilized in Neural Network **530***a*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, and/or other data structures or arrangements. In some aspects, instead of anticipating Instruction Sets **526** correlated with substantially similar individual Digital Pictures **525**, Decision-making Unit **540** can anticipate instruction Sets **526** correlated with substantially matching streams of Digital Pictures **525**. In other aspects, any time that substantial similarity or other similarity threshold is not achieved in compared Digital Pictures **525** or portions thereof of any of the Knowledge Cells **800**, Decision-making Unit **540** can decide to look for a substantially or otherwise similar Digital Pictures **525** or portions thereof in Knowledge Cells **800** elsewhere in Graph **530***b* even if such Knowledge Cell **800** may be unconnected with a prior Knowledge Cell **800**. It should be noted that any of Digital Pictures **525***a***1**-**525***an*, Digital Pictures **525***b***1**-**525***bn*, Digital Pictures **525***c***1**-**525***cn*, Digital Pictures **525***d***1**-**525***dn*, Digital Pictures **525***e***1**-**525***en*, etc. may include one Digital Picture **525** or a stream of Digital Pictures **525**. It should also be noted that any Knowledge Cell **800** may include one Digital Picture **525** or a stream of Digital Pictures **525** as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to FIG. **28**, an embodiment of determining anticipatory Instruction Sets **526** using Collection of Sequences **530***c* is illustrated. Collection of Sequences **530***c* may include knowledge (i.e. sequences of Knowledge Cells **800** comprising one or more Digital Pictures **525** correlated with any Instruction Sets **526** and/or Extra Info **527**, etc.) of how Device **98** operated in various visual surroundings. In some aspects, determining anticipatory Instruction Sets **526** for autonomous Device **98** operation using Collection of Sequences **530***c* may include selecting a Sequence **533** of Knowledge Cells **800** or elements (i.e. Digital Pictures **525**, Instruction Sets **526**, etc.) thereof from Collection of Sequences **530***c*. Individual and/or collective Similarity Comparisons **125** can be used to determine substantial similarity of the individually and/or collectively compared Digital Pictures **525** or portions thereof.

For example, Decision-making Unit **540** can perform Similarity Comparisons **125** of Digital Pictures **525***a***1**-**525***an* or portions thereof from Picture Capturing Apparatus **90** with Digital Pictures **525** or portions thereof from corresponding Knowledge Cells **800** in one or more Sequences **533** of Collection of Sequences **530***c*. Digital Pictures **525** or portions thereof from Knowledge Cell **800***ca* in Sequence **533***wc* may be found collectively substantially similar with highest similarity. As the comparisons of individual Digital Pictures **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Digital Pictures **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Digital Pictures **525***a***1**-**525***an* and **525***b***1**-**525***bn* or portions thereof from Picture Capturing Apparatus **90** with

Digital Pictures **525** or portions thereof from corresponding Knowledge Cells **800** in Sequences **533** of Collection of Sequences **530***c*. Digital Pictures **525** or portions thereof from Knowledge Cells **800***ca*-**800***cb* in Sequence **533***wc* may be found collectively substantially similar with highest similarity. As the comparisons of individual Digital Pictures **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Digital Pictures **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Digital Pictures **525***a***1**-**525***an*, **525***b***1**-**525***bn*, and **525***c***1**-**525***cn* or portions thereof from Picture Capturing Apparatus **90** with Digital Pictures **525** or portions thereof from corresponding Knowledge Cells **800** in Sequences **533** of Collection of Sequences **530***c*. Digital Pictures **525** or portions thereof from Knowledge Cells **800***da*-**800***dc* in Sequence **533***wd* may be found substantially similar with highest similarity. As the comparisons of individual Digital Pictures **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Digital Pictures **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Digital Pictures **525***a***1**-**525***an*, **525***b***1**-**525***bn*, **525***c***1**-**525***cn*, and **525***d***1**-**525***dn* or portions thereof from Picture Capturing Apparatus **90** with Digital Pictures **525** or portions thereof from corresponding Knowledge Cells **800** in Sequences **533** of Collection of Sequences **530***c*. Digital Pictures **525** or portions thereof from Knowledge Cells **800***da*-**800***dd* in Sequence **533***wd* may be found substantially similar with highest similarity. As the comparisons of individual Digital Pictures **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Digital Pictures **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can then perform Similarity Comparisons **125** of Digital Pictures **525***a***1**-**525***an*, **525***b***1**-**525***bn*, **525***c***1**-**525***cn*, **525***d***1**-**525***dn*, and **525***e***1**-**525***en* or portions thereof from Picture Capturing Apparatus **90** with Digital Pictures **525** or portions thereof from corresponding Knowledge Cells **800** in Sequences **533** of Collection of Sequences **530***c*. Digital Pictures **525** or portions thereof from Knowledge Cells **800***da*-**800***de* in Sequence **533***wd* may be found substantially similar with highest similarity. As the comparisons of individual Digital Pictures **525** are performed to determine collective similarity, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially similar individual Digital Pictures **525** as previously described, thereby enabling autonomous Device **98** operation. Decision-making Unit **540** can implement similar logic or process for any additional Digital Pictures **525** from Picture Capturing Apparatus **90**, and so on.

In some embodiments, various elements and/or techniques can be utilized in the aforementioned substantial similarity determinations with respect to collectively compared Knowledge Cells **800** or elements (i.e. Digital Pictures **525**, Extra Info **527**, etc.) thereof. In some aspects, substantial similarity of collectively compared Knowledge Cells **800** or elements (i.e. Digital Pictures **525**, Extra Info **527**, etc.) thereof can be determined based on similarities or similarity indexes of the individually compared Knowledge Cells **800** or elements (i.e. Digital Pictures **525**, Extra Info **527**, etc.) thereof. In one example, an average of similarities or similarity indexes of individually compared Knowledge Cells **800** or elements (i.e. Digital Pictures **525**, Extra Info **527**, etc.) thereof can be used to determine similarity of collectively compared Knowledge Cells **800** or elements (i.e. Digital Pictures **525**, Extra Info **527**, etc.) thereof. In another example, a weighted average of similarities or similarity indexes of individually compared Knowledge Cells **800** or elements (i.e. Digital Pictures **525**, Extra Info **527**, etc.) thereof can be used to determine similarity of collectively compared Knowledge Cells **800** or elements (i.e. Digital Pictures **525**, Extra Info **527**, etc.) thereof. For instance, to affect the weighting of collective similarity, a higher weight or importance (i.e. importance index, etc.) can be assigned to the similarities or similarity indexes of some Knowledge Cells **800** or elements (i.e. Digital Pictures **525**, Extra Info **527**, etc.) thereof and lower for other Knowledge Cells **800** or elements (i.e. Digital Pictures **525**, Extra Info **527**, etc.) thereof. Any higher or lower weight or importance assignment can be implemented. In other aspects, any of the previously described or other thresholds for substantial similarity of individually compared elements can similarly be utilized for collectively compared elements. In one example, substantial similarity of collectively compared Knowledge Cells **800** or elements (i.e. Digital Pictures **525**, Extra Info **527**, etc.) thereof can be achieved when their collective similarity or similarity index exceeds a similarity threshold. In another example, substantial similarity of collectively compared Knowledge Cells **800** can be achieved when at least a threshold number or percentage of Digital Pictures **525** or portions thereof of the collectively compared Knowledge Cells **800** match or substantially match. Similarly, substantial similarity of collectively compared Knowledge Cells **800** can be achieved when a number or percentage of matching or substantially matching Digital Pictures **525** or portions thereof of the collectively compared Knowledge Cells **800** exceeds a threshold. Such thresholds can be defined by a user, by VSADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. Collective similarity determinations may include any features, functionalities, and embodiments of Similarity Comparison **125**, and vice versa.

The foregoing exemplary embodiment provides an example of utilizing a combination of collective Similarity Comparisons **125**, individual Similarity Comparisons **125**, and/or other elements or techniques. It should be understood that any of these elements and/or techniques can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Sequence **533** of Knowledge Cells **800** or elements (i.e. Digital Pictures **525**, Instruction Sets **526**, etc.) thereof would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info **527** in Similarity Comparisons **125**, traversing of Knowledge Cells **800** or other elements, using history of Digital Pictures **525** or Knowledge Cells **800** in collective Similarity Comparisons **125**, using various arrangements of Digital Pictures **525** and/or other elements in a Knowledge Cell **800**, and/or others can similarly be utilized in this example. These elements and/or techniques can similarly be utilized in Neural Network **530***a*, Graph **530***b*, Collection of Knowledge Cells **530***d*, and/or other data structures or arrangements. In some aspects, instead of anticipating Instruction Sets **526** correlated with substantially similar individual Digital Pictures **525**, Decision-making Unit **540** can anticipate Instruction Sets **526** correlated with substantially matching streams of Digital Pictures **525**. In other aspects,

any time that substantial similarity or other similarity threshold is not achieved in compared Digital Pictures **525** or portions thereof of any of the Knowledge Cells **800**, Decision-making Unit **540** can decide to look for a substantially or otherwise similar Digital Pictures **525** or portions thereof in Knowledge Cells **800** elsewhere in Collection of Sequences **530***c* such as in different Sequences **533**. It should be noted that any of Digital Pictures **525***a***1**-**525***an*, Digital Pictures **525***b***1**-**525***bn*, Digital Pictures **525***c***1**-**525***cn*, Digital Pictures **525***d***1**-**525***dn*, Digital Pictures **525***e***1**-**525***en*, etc. may include one Digital Picture **525** or a stream of Digital Pictures **525**. It should also be noted that any Knowledge Cell **800** may include one Digital Picture **525** or a stream of Digital Pictures **525** as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring now to Modification Interface **130**. Modification Interface **130** comprises the functionality for modifying execution and/or functionality of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. Modification Interface **130** comprises the functionality for modifying execution and/or functionality of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element at runtime. Modification Interface **130** comprises the functionality for modifying execution and/or functionality of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element based on anticipatory Instruction Sets **526**. In one example, Modification Interface **130** comprises the functionality to access, modify, and/or perform other manipulations on runtime engine/environment, virtual machine, operating system, compiler, just-in-time (JIT) compiler, interpreter, translator, execution stack, file, object, data structure, and/or other computing system elements. In another example, Modification Interface **130** comprises the functionality to access, modify, and/or perform other manipulations on memory, storage, bus, interfaces, and/or other computing system elements. In a further example, Modification Interface **130** comprises the functionality to access, modify, and/or perform other manipulations on Processor **11** registers and/or other Processor **11** elements. In a further example, Modification Interface **130** comprises the functionality to access, modify, and/or perform other manipulations on inputs and/or outputs of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. In a further example, Modification Interface **130** comprises the functionality to access, create, delete, modify, and/or perform other manipulations on functions, methods, procedures, routines, subroutines, and/or other elements of Application Program **18**. In a further example, Modification Interface **130** comprises the functionality to access, create, delete, modify, and/or perform other manipulations on source code, bytecode, compiled, interpreted, or otherwise translated code, machine code, and/or other code. In a further example, Modification Interface **130** comprises the functionality to access, create, delete, modify, and/or perform other manipulations on values, variables, parameters, and/or other data or information. Modification Interface **130** comprises any features, functionalities, and embodiments of Acquisition Interface **120**, and vice versa. Modification Interface **130** also comprises other disclosed functionalities.

Modification Interface **130** can employ various techniques for modifying execution and/or functionality of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. In some aspects, some of the previously described techniques and/or tools can be utilized. Code instrumentation, for instance, may involve inserting additional code, overwriting or rewriting existing code, and/or branching to a separate segment of code in Application Program **18** as previously described. For example, instrumented code may include the following:

```
Object1.moveLeft(12);
modifyApplication( );
```

In the above sample code, instrumented call to Modification Interface's **130** function (i.e. modifyApplication( ), etc.) can be placed after a function (i.e. moveLeft(12), etc.) of Application Program **18**. Similar call to an application modifying function can be placed after or before some or all functions/routines/subroutines, some or all lines of code, some or all statements, some or all instructions or instruction sets, some or all basic blocks, and/or some or all other code segments of Application Program **18**. One or more application modifying function calls can be placed anywhere in Application Program's **18** code and can be executed at any points in Application Program's **18** execution. The application modifying function (i.e. modifyApplication( ), etc.) may include Artificial Intelligence Unit **110**-determined anticipatory Instruction Sets **526** that can modify execution and/or functionality of Application Program **18**. In some embodiments, the previously described obtaining Application Program's **18** instruction sets, data, and/or other information as well as modifying execution and/or functionality of Application Program **18** can be implemented in a single function that performs both tasks (i.e. traceAndModifyApplication( ), etc.).

In some embodiments, various computing systems and/or platforms may provide native tools for modifying execution and/or functionality of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element. Independent vendors may provide tools with similar functionalities that can be utilized across different platforms. These tools enable a wide range of techniques or capabilities such as instrumentation, self-modifying code capabilities, dynamic code capabilities, branching, code rewriting, code overwriting, hot swapping, accessing and/or modifying objects or data structures, accessing and/or modifying functions/routines/subroutines, accessing and/or modifying variable or parameter values, accessing and/or modifying processor registers, accessing and/or modifying inputs and/or outputs, providing runtime memory access, and/or other capabilities. One of ordinary skill in art will understand that, while all possible variations of the techniques for modifying execution and/or functionality of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing element are too voluminous to describe, these techniques are within the scope of this disclosure.

In one example, modifying execution and/or functionality of Application Program **18** can be implemented through utilizing metaprogramming techniques, which include applications that can self-modify or that can create, modify, and/or manipulate other applications. Self-modifying code, dynamic code, reflection, and/or other techniques can be used to facilitate metaprogramming. In some aspects, metaprogramming is facilitated through a programming language's ability to access and manipulate the internals of the runtime engine directly or via an API. In other aspects, metaprogramming is facilitated through dynamic execution of expressions (i.e. anticipatory Instruction Sets **526**, etc.) that can be created and/or executed at runtime. In yet other aspects, metaprogramming is facilitated through application modification tools, which can perform modifications on an

application regardless of whether the application's programming language enables metaprogramming capabilities. Some operating systems may protect an application loaded into memory by restricting access to the loaded application. This protection mechanism can be circumvented by utilizing operating system's, processor's, and/or other low level features or commands to unprotect the loaded application. For example, a self-modifying application may modify the in-memory image of itself. To do so, the application can obtain the in-memory address of its code. The application may then change the operating system's or platform's protection on this memory range allowing it to modify the code (i.e. insert anticipatory Instruction Sets **526**, etc.). In addition to a self-modifying application, one application can utilize similar technique to modify another application. Linux mprotect command or similar commends of other operating systems can be used to change protection (i.e. unprotect, etc.) for a region of memory, for example. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program **18** can be implemented through native capabilities of dynamic, interpreted, and/or scripting programming languages and/or platforms. Most of these languages and/or platforms can perform functionalities at runtime that static programming languages may perform during compilation. Dynamic, interpreted, and/or scripting languages provide native functionalities such as self-modification of code, dynamic code, extending the application, adding new code, extending objects and definitions, and/or other functionalities that can modify an application's execution and/or functionality at runtime. Examples of dynamic, interpreted, and/or scripting languages include Lisp, Perl, PHP, JavaScript, Ruby, Python, Smalltalk, Tcl, VBScript, and/or others. Similar functionalities can also be provided in languages such as Java, C, and/or others using reflection. Reflection includes the ability of an application to examine and modify the structure and behavior of the application at runtime. For example, JavaScript can modify its own code as it runs by utilizing Function object constructor as follows:

```
myFunc=new Function(arg1, arg2, argN, functionBody);
```

The sample code above causes a new function object to be created with the specified arguments and body. The body and/or arguments of the new function object may include new instruction sets (i.e. anticipatory Instruction Sets **526**, etc.). The new function can be invoked as any other function in the original code. In another example, JavaScript can utilize eval method that accepts a string of JavaScript statements (i.e. anticipatory Instruction Sets **526**, etc.) and execute them as if they were within the original code. An example of how eval method can be used to modify an application includes the following JavaScript code:

```
anticipatoryInstr='Object1.moveForward(32);';
if (anticipatoryInstr !=" " && anticipatoryInstr !=null)
{
eval(anticipatoryInstr);
}
```

In the sample code above, Artificial Intelligence Unit **110** may generate anticipatory Instruction Set **526** (i.e. 'Object1.moveForward(32)' for moving an object forward 32 units, etc.) and save it in anticipatoryInstr variable, which eval method can then execute. Lisp is another example of dynamic, interpreted, and/or scripting language that includes similar capabilities as previously described JavaScript. For example, Lisp's compile command can create a function at runtime, eval command may parse and evaluate an expression at runtime, and exec command may execute a given instruction set (i.e. string, etc.) at runtime. In another example, dynamic as well as some non-dynamic languages may provide macros, which combine code introspection and/or eval capabilities. In some aspects, macros can access inner workings of the compiler, interpreter, virtual machine, runtime environment/engine, and/or other components of the computing platform enabling the definition of language-like constructs and/or generation of a complete program or sections thereof. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program **18** can be implemented through dynamic code, dynamic class loading, reflection, and/or other native functionalities of a programming language or platform. In static applications or static programming, a class can be defined and/or loaded at compile time. Conversely, in dynamic applications or dynamic programming, a class can be loaded into a running environment at runtime. For example, Java Runtime Environment (JRE) may not require that all classes be loaded at compile time and class loading can occur when a class is first referenced at runtime. Dynamic class loading enables inclusion or injection of on-demand code and/or functionalities at runtime. System provided or custom class loaders may enable loading of classes into the running environment. Custom class loaders can be created to enable custom functionalities such as, for example, specifying a remote location from which a class can be loaded. In addition to dynamic loading of a pre-defined class, a class can also be created at runtime. In some aspects, a class source code can be created at runtime. A compiler such as javac, com.sun.tools.javac.Main, javax.tools, javax.tools.JavaCompiler, and/or other packages can then be utilized to compile the source code. Javac, com.sun.tools.javac.Main, javax.tools, javax.tools.JavaCompiler, and/or other packages may include an interface to invoke Java compiler from within a running application. A Java compiler may accept source code in a file, string, object (i.e. Java String, StringBuffer, CharSequence, etc.) and/or other source, and may generate Java bytecode (i.e. class file, etc.). Once compiled, a class loader can then load the compiled class into the running environment. In other aspects, a tool such as Javaassist (i.e. Java programming assistant) can be utilized to enable an application to create or modify a class at runtime. Javassist may include a Java library that provides functionalities to create and/or manipulate Java bytecode of an application as well as reflection capabilities. Javassist may provide source-level and bytecode-level APIs. Using the source-level API, a class can be created and/or modified using only source code, which Javassist may compile seamlessly on the fly. Javassist source-level API can therefore be used without knowledge of Java bytecode specification. Bytecode-level API enables creating and/or editing a class bytecode directly. In yet other aspects, similar functionalities to the aforementioned ones may be provided in tools such as Apache Commons BCEL (Byte Code Engineering Library), ObjectWeb ASM, CGLIB (Byte Code Generation Library), and/or others. Once a dynamic code or class is created and loaded, reflection in high-level programming languages such as Java and/or others can be used to manipulate or change the runtime behavior of an application. Examples of reflective programming languages and/or platforms include Java, JavaScript, Smalltalk, Lisp, Python, .NET Common Language Runtime (CLR), Tcl, Ruby, Perl, PHP, Scheme, PL/SQL, and/or others. Reflection can be used in an application to access, examine, modify, and/or manipulate a loaded class and/or its

elements. Reflection in Java can be implemented by utilizing a reflection API such as java.lang.Reflect package. The reflection API provides functionalities such as, for example, loading or reloading a class, instantiating a new instance of a class, determining class and instance methods, invoking class and instance methods, accessing and manipulating a class, fields, methods and constructors, determining the modifiers for fields, methods, classes, and interfaces, and/or other functionalities. The above described dynamic code, dynamic class loading, reflection, and/or other functionalities are similarly provided in the .NET platform through its tools such as, for example, System.CodeDom.Compiler namespace, System.Reflection.Emit namespace, and/or other native or other .NET tools. Other platforms in addition to Java and .NET may provide similar tools and/or functionalities. In some designs, dynamic code, dynamic class loading, reflection, and/or other functionalities can be used to facilitate modification of an application by inserting or injecting instruction sets (i.e. anticipatory Instruction Sets **526**, etc.) into a running application. For example, an existing or dynamically created class comprising VSADO Unit **100** functionalities can be loaded into a running application through manual, automatic, or dynamic instrumentation. Once the class is created and loaded, an instance of VSADO Unit **100** class may be constructed. The instance of VSADO Unit **100** can then take or exert control of the application and/or implement alternate instruction sets (i.e. anticipatory Instruction Sets **526**, etc.) at any point in the application's execution. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program **18** can be implemented through independent tools that can be utilized across different platforms. Such tools provide instrumentation and/or other capabilities on more than one platform or computing system and may facilitate application modification or insertion of instruction sets (i.e. anticipatory Instruction Sets **526**, etc.). Examples of these tools include Pin, DynamoRIO, DynInst, Kprobes, KernInst, OpenPAT, DTrace, SystemTap, and/or others. In some aspects, Pin and/or any of its elements, methods, and/or techniques can be utilized for dynamic instrumentation. Pin can perform instrumentation by taking control of an application after it loads into memory. Pin may insert itself into the address space of an executing application enabling it to take control. Pin JIT compiler can then compile and implement alternate code (i.e. anticipatory Instruction Sets **526**, etc.). Pin provides an extensive API for instrumentation at several abstraction levels. Pin supports two modes of instrumentation, JIT mode and probe mode. JIT mode uses a just-in-time compiler to insert instrumentation and recompile program code while probe mode uses code trampolines for instrumentation. Pin was designed for architecture and operating system independence. In other aspects, KernInst and/or any of its elements, methods, and/or techniques can be utilized for dynamic instrumentation. KernInst includes an instrumentation framework designed for dynamically inserting code into a running kernel of an operating system. KernInst implements probe-based dynamic instrumentation where code can be inserted, changed, and/or removed at will. Kerninst API enables client tools to construct their own tools for dynamic kernel instrumentation to suit variety of purposes such as insertion of alternate instruction sets (i.e. anticipatory Instruction Sets **526**, etc.). Client tools can communicate with KernInst over a network (i.e. internet, wireless network, LAW, WAN, etc). Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program **18** can be implemented through utilizing operating system's native tools or capabilities such as Unix ptrace command. Ptrace includes a system call that may enable one process to control another allowing the controller to inspect and manipulate the internal state of its target. Ptrace can be used to modify a running application such as modifying an application with alternate instruction sets (i.e. anticipatory Instruction Sets **526**, etc.). By attaching to an application using the ptrace call, the controlling application can gain extensive control over the operation of its target. This may include manipulation of its instruction sets, execution path, file descriptors, memory, registers, and/or other components. Ptrace can single-step through the target's code, observe and intercept system calls and their results, manipulate the target's signal handlers, receive and send signals on the target's behalf, and/or perform other operations within the target application. Ptrace's ability to write into the target application's memory space enables the controller to modify the running code of the target application. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program **18** can be implemented through utilizing just-in-time (JIT) compiling. JIT compilation (also known as dynamic translation, dynamic compilation, etc.) includes compilation performed during an application's execution (i.e. runtime, etc.). A code can be compiled when it is about to be executed, and it may be cached and reused later without the need for additional compilation. In some aspects, a JIT compiler can convert source code or byte code into machine code. In other aspects, a JIT compiler can convert source code into byte code. JIT compiling may be performed directly in memory. For example, JIT compiler can output machine code directly into memory and immediately execute it. Platforms such as Java, .NET, and/or others may implement JIT compilation as their native functionality. Platform independent tools for custom system design may include JIT compilation functionalities as well. In some aspects, JIT compilation includes redirecting application's execution to a JIT compiler from a specific entry point. For example, Pin can insert its JIT compiler into the address space of an application. Once execution is redirected to it, JIT compiler may receive alternate instruction sets (i.e. anticipatory Instruction Sets **526**, etc.) immediately before their compilation. The JIT compiled instruction sets can be stored in memory or another repository from where they may be retrieved and executed. Alternatively, for example, JIT compiler can create a copy of the original application code or a segment thereof, and insert alternate code (i.e. anticipatory Instruction Sets **526**, etc.) before compiling the modified code copy. In some aspects, JIT compiler may include a specialized memory such as fast cache memory dedicated to JIT compiler functionalities from which the modified code can be fetched rapidly. JIT compilation and/or any compilation in general may include compilation, interpretation, or other translation into machine code, bytecode, and/or other formats or types of code. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program **18** can be implemented through dynamic recompilation. Dynamic recompilation

includes recompiling an application or part thereof during execution. An application can be modified with alternate features or instruction sets that may take effect after recompilation. Dynamic recompilation may be practical in various types of applications including object oriented, event driven, forms based, and/or other applications. In a typical windows-based application, most of the action after initial startup occurs in response to user or system events such as moving the mouse, selecting a menu option, typing text, running a scheduled task, making a network connection, and/or other events when an event handler is called to perform an operation appropriate for the event. Generally, when no events are being generated, the application is idle. For example, when an event occurs and an appropriate event handler is called, instrumentation can be implemented in the application's source code to insert alternate instruction sets (i.e. anticipatory Instruction Sets **526**, etc.) at which point the modified source code can be recompiled and/or executed. In some aspects, the state of the application can be saved before recompiling its modified source code so that the application may continue from its prior state. Saving the application's state can be achieved by saving its variables, data structures, objects, location of its current instruction, and/or other necessary information in environmental variables, memory, or other repositories where they can be accessed once the application is recompiled. In other aspects, application's variables, data structures, objects, address of its current instruction, and/or other necessary information can be saved in a repository such as file, database, or other repository accessible to the application after recompilation of its source code. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program **18** can be implemented through modifying or redirecting Application Program's **18** execution path. Generally, an application can be loaded into memory and the flow of execution proceeds from one instruction set to the next until the end of the application. An application may include a branching mechanism that can be driven by keyboard or other input devices, system events, and/or other computing system components or events that may impact the execution path. The execution path can also be altered by an external application through acquiring control of execution and/or redirecting execution to a function, routine/subroutine, or an alternate code segment at any point in the application's execution. A branch, jump, or other mechanism can be utilized to implement the redirected execution. For example, a jump instruction can be inserted at a specific point in an application's execution to redirect execution to an alternate code segment. A jump instruction set may include, for example, an unconditional branch, which always results in branching, or a conditional branch, which may or may not result in branching depending on a condition. When executing an application, a computer may fetch and execute instruction sets in sequence until it encounters a branch instruction set. If the instruction set is an unconditional branch, or it is conditional and the condition is satisfied, the computer may fetch its next instruction set from a different instruction set sequence or code segment as specified by the branch instruction set. After the execution of the alternate code segment, control may be redirected back to the original jump point or to another point in the application. For example, modifying an application can be implemented by redirecting execution of an application to alternate instruction sets (i.e. anticipatory Instruction Sets **526**, etc.). Alternate instruction sets can be pre-compiled, pre-interpreted, or otherwise pre-translated and ready for execution. Alternate instruction sets can also be JIT compiled, JIT interpreted, or otherwise JIT translated before execution. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program **18** can be implemented through assembly language. Assembly language instructions may be directly related with the architecture's machine instructions as previously described. Assembly language can, therefore, be a powerful tool for implementing direct hardware (i.e. processor registers, memory, etc.) access and manipulations as well as access and manipulations of specialized processor features or instructions. Assembly language can also be a powerful tool for implementing low-level embedded systems, real-time systems, interrupt handlers, self or dynamically modifying code, and/or other applications. Specifically, for instance, self or dynamically modifying code that can be used to facilitate modifying of an application can be seamlessly implemented using assembly language. For example, using assembly language, instruction sets can be dynamically created and loaded into memory similar to the ones that a compiler may generate. Furthermore, using assembly language, memory space of a loaded application can be accessed to modify (including rewrite, overwrite, etc.) original instruction sets or to insert jumps or branches to alternate code elsewhere in memory. Some operating systems may implement protection from changes to applications loaded into memory. Operating system's, processor's, or other low level features or commands can be used to unprotect the protected locations in memory before the change as previously described. Alternatively, a pointer that may reside in a memory location where it could be readily altered can be utilized where the pointer may reference alternate code. In one example, assembly language can be utilized to write alternate code (i.e. anticipatory Instruction Sets **526**, etc.) into a location in memory outside a running application's memory space. Assembly language can then be utilized to redirect the application's execution to the alternate code by inserting a jump or branch into the application's in-memory code, by redirecting program counter, or by other technique. In another example, assembly language can be utilized to overwrite or rewrite the entire or part of an application's in-memory code with alternate code. In some aspects, high-level programming languages can call an external assembly language program to facilitate application modification as previously described. In yet other aspects, relatively low-level programming languages such as C may allow embedding assembly language directly in their source code such as, for example, using asm keyword of C. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program **18** can be implemented through binary rewriting. Binary rewriting tools and/or techniques may modify an application's executable. In some aspects, modification can be minor such as in the case of optimization where the original executable's functionality is kept. In other aspects, modification may change the application's functionality such as by inserting alternate code (i.e. anticipatory Instruction Sets **526**, etc.). Examples of binary rewriting tools include SecondWrite, ATOM, DynamoRIO, Purify, Pin, EEL, DynInst, PLTO, and/or others. Binary rewriting may include disassembly, analysis, and/or modification of target application. Since binary rewriting works

directly on machine code executable, it is independent of source language, compiler, virtual machine (if one is utilized), and/or other higher level abstraction layers. Also, binary rewriting tools can perform application modifications without access to original source code. Binary rewriting tools include static rewriters, dynamic rewriters, minimally-invasive rewriters, and/or others. Static binary rewriters can modify an executable when the executable is not in use (i.e. not running). The rewritten executable may then be executed including any new or modified functionality. Dynamic binary rewriters can modify an executable during its execution, thereby enabling modification of an application's functionality at runtime. In some aspects, dynamic rewriters can be used for instrumentation or selective modifications such as insertion of alternate code (i.e. anticipatory Instruction Sets **526**, etc.), and/or for other runtime transformations or modifications. For example, some dynamic rewriters can be configured to intercept an application's execution at indirect control transfers and insert instrumentation or other application modifying code. Minimally-invasive rewriters may keep the original machine code to the greatest extent possible. They support limited modifications such as insertion of jumps into and out of instrumented code. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

Referring to FIG. **29**, in a further example, modifying execution and/or functionality of Processor **11** can be implemented through modification of processor registers, memory, or other computing system components. In some aspects, modifying execution and/or functionality of Processor **11** can be implemented by redirecting Processor's **11** execution to alternate instruction sets (i.e. anticipatory Instruction Sets **526**, etc.). In one example, Program Counter **211** may hold or point to a memory address of the next instruction set that will be executed by Processor **11**. Artificial Intelligence Unit **110** may generate anticipatory Instruction Sets **526** and store them in Memory **12** as previously described. Modification Interface **130** may then change Program Counter **211** to point to the location in Memory **12** where anticipatory Instruction Sets **526** are stored. The anticipatory Instruction Sets **526** can then be fetched from the location in Memory **12** pointed to by the modified Program Counter **211** and loaded into Instruction Register **212** for decoding and execution. Once anticipatory Instruction Sets **526** are executed, Modification Interface **130** may change Program Counter **211** to point to the last instruction set before the redirection or to any other instruction set. In other aspects, anticipatory Instruction Sets **526** can be loaded directly into Instruction Register **212**. As previously described, examples of other processor or computing system components that can be used during an instruction cycle include memory address register (MAR), memory data register (MDR), data registers, address registers, general purpose registers (GPRs), conditional registers, floating point registers (FPRs), constant registers, special purpose registers, machine-specific registers, Register Array **214**, Arithmetic Logic Unit **215**, control unit, and/or other circuits or components. Any of the aforementioned processor registers, memory, or other computing system components can be accessed and/or modified to facilitate the disclosed functionalities. In some embodiments, processor interrupt may be issued to facilitate such access and/or modification. In some designs, modifying execution and/or functionality of Processor **11** can be implemented in a program, combination of programs and hardware, or purely hardware system. Dedicated hardware may be built to perform modifying execution and/or functionality of Processor **11** with marginal or no impact to computing overhead. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

Referring to FIGS. **30**A-**30**B, in a further example, modifying execution and/or functionality of Logic Circuit **250** can be implemented through modification of inputs and/or outputs of Logic Circuit **250**. While Processor **11** includes any type of logic circuit, Logic Circuit **250** is described separately herein to offer additional detail on its functioning. Logic Circuit **250** comprises the functionality for performing logic operations using the circuit's inputs and producing outputs based on the logic operations performed as previously described. In one example, Logic Circuit **250** may perform some logic operations using four input values and produce two output values. Modifying execution and/or functionality of Logic Circuit **250** can be implemented by replacing its input values with anticipatory input values (i.e. anticipatory Instruction Sets **526**, etc.). Artificial Intelligence Unit **110** may generate anticipatory input values as previously described. Modification Interface **130** can then transmit the anticipatory input values to Logic Circuit **250** through the four hardwired connections as shown in FIG. **30**A. Modification Interface **130** may use Switches **251** to prevent delivery of any input values that may be sent to Logic Circuit **250** from its usual input source. As such, VSADO Unit **100** may cause Logic Circuit **250** to perform its logic operations using the four anticipatory input values, thereby implementing autonomous Device **98** operation. In another example, Logic Circuit **250** may perform some logic operations using four input values and produce two output values. Modifying execution and/or functionality of Logic Circuit **250** can be implemented by replacing its output values with anticipatory output values (i.e. anticipatory Instruction Sets **526**, etc.). Artificial Intelligence Unit **110** may generate anticipatory output values (i.e. anticipatory Instruction Sets **526**, etc.) as previously described. Modification Interface **130** can then transmit the anticipatory output values through the two hardwired connections as shown in FIG. **30**B. Modification Interface **130** may use Switches **251** to prevent delivery of any output values that may be sent by Logic Circuit **250**. As such, VSADO Unit **100** may bypass Logic Circuit **250** and transmit the two anticipatory output values to downstream elements, thereby implementing autonomous Device **98** operation. In a further example, instead of or in addition to modifying input and/or output values of Logic Circuit **250**, the execution and/or functionality of Logic Circuit **250** may be modified by modifying values or signals in one or more Logic Circuit's **250** internal components such as registers, memories, buses, and/or others (i.e. similar to the previously described modifying of Processor **11** components, etc.). In some designs, modifying execution and/or functionality of Logic Circuit **250** can be implemented in a program, combination of programs and hardware, or purely hardware system. Dedicated hardware may be built to perform modifying execution and/or functionality of Logic Circuit **250** with marginal or no impact to computing overhead. Any of the elements and/or techniques for modifying execution and/or functionality of Logic Circuit **250** can similarly be implemented with Processor **11** and/or other processing elements.

In some embodiments, VSADO Unit **100** may directly modify the functionality of an actuator (previously described, not shown). For example, Logic Circuit **250** or other processing element may control an actuator that enables Device **98** to perform mechanical, physical, and/or other operations. An actuator may receive one or more input values or control signals from Logic Circuit **250** or other

processing element directing the actuator to perform specific operations. Modifying functionality of an actuator can be implemented by replacing its input values with anticipatory input values (i.e. anticipatory Instruction Sets **526**, etc.) as previously described with respect to replacing input values of Logic Circuit **250**. Specifically, for instance, Artificial Intelligence Unit **110** may generate anticipatory input values as previously described. Modification Interface **130** can then transmit the anticipatory input values to the actuator. Modification Interface **130** may use Switches **251** to prevent delivery of any input values that may be sent to the actuator from its usual input source. As such, VSADO Unit **100** may cause the actuator to perform its operations using the anticipatory input values, thereby implementing autonomous Device **98** operation.

One of ordinary skill in art will recognize that FIGS. **30**A-**30**B depict one of many implementations of Logic Circuit **250** and that any number of input and/or output values can be utilized in alternate implementations. One of ordinary skill in art will also recognize that Logic Circuit **250** may include any number and/or combination of logic components to implement any logic operations.

Other additional techniques or elements can be utilized as needed for modifying execution and/or functionality of Application Program **18**, Processor **11**, Logic Circuit **250**, and/or other processing elements, or some of the disclosed techniques or elements can be excluded, or a combination thereof can be utilized in alternate embodiments.

Referring to FIG. **31**, the illustration shows an embodiment of a method **6100** for learning and/or using visual surrounding for autonomous device operation. The method can be used on a computing device or system to enable learning of a device's operation in various visual surroundings and enable autonomous device operation in similar visual surroundings. Method **6100** may include any action or operation of any of the disclosed methods such as method **6200**, **6300**, **6400**, **6500**, **6600**, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method **6100**.

At step **6105**, a first digital picture is received. A digital picture (i.e. Digital Picture **525**, etc.) may include a depiction of a device's (i.e. Device's **98**, etc.) visual surrounding. A digital picture may include a depiction of a remote device's (i.e. Remote Device's **97**, etc.) visual surrounding. In some embodiments, a digital picture may include a collection of color encoded pixels or dots. A digital picture comprises any type or form of digital picture such as JPEG, GIF, TIFF, PNG, PDF, and/or other digitally encoded picture. In other embodiments, a stream of digital pictures (i.e. motion picture, video, etc.) may include one or more digital pictures. A stream of digital pictures comprises any type or form of digital motion picture such as MPEG, AVI, FLV, MOV, RM, SWF, WMV, DivX, and/or other digitally encoded motion picture. In some aspects, a digital picture may include or be substituted with a stream of digital pictures, and vice versa. Therefore, the terms digital picture and stream of digital pictures may be used interchangeably herein depending on context. One or more digital pictures can be captured by a picture capturing apparatus (i.e. Picture Capturing Apparatus **90**, etc.) such as a still or motion picture camera, or other picture capturing apparatus. In some aspects, a picture capturing apparatus may be part of a device whose visual surrounding is being used for VSADO functionalities. In other aspects, a picture capturing apparatus may be part of a remote device, accessible via a network, whose visual surrounding is being used for VSADO functionalities. Picture capturing apparatus may be provided in any other device, system, process, or configuration. In some embodiments, capturing and/or receiving may be responsive to a triggering object, action, event, time, and/or other stimulus. Receiving comprises any action or operation by or for a Picture Capturing Apparatus **90**, Digital Picture **525**, and/or other disclosed elements.

At step **6110**, one or more instruction sets for operating a device are received. In some embodiments, an instruction set (i.e. Instruction Set **526**, etc.) may be used or executed by a processor (i.e. Processor **11**, etc.) for operating a device (i.e. Device **98**, etc.). In other embodiments, an instruction set may be part of an application program (i.e. Application Program **18**, etc.) for operating a device. The application can run or execute on one or more processors or other processing elements. In further embodiments, an instruction set may be used, executed, or produced by a logic circuit (i.e. Logic Circuit **250**, etc.) for operating a device. For example, such instruction set may be or include one or more inputs into or outputs from a logic circuit. In further embodiments, an instruction set may be used by an actuator for operating a device. For example, such instruction set may be or include one or more inputs into an actuator. Operating a device includes performing any operations on or with the device. An instruction set may temporally correspond to a digital picture. In some aspects, an instruction set that temporally corresponds to a digital picture may include an instruction set used or executed at the time of receiving or capturing the digital picture. In other aspects, an instruction set that temporally corresponds to a digital picture may include an instruction set used or executed within a certain time period before and/or after receiving or capturing the digital picture. Any time period may be utilized. In further aspects, an instruction set that temporally corresponds to a digital picture may include an instruction set used or executed from the time of capturing of the digital picture to the time of capturing of a next digital picture. In further aspects, an instruction set that temporally corresponds to a digital picture may include an instruction set used or executed from the time of capturing of a preceding digital picture to the time of capturing of the digital picture. Any other temporal relationship or correspondence between digital pictures and correlated instruction sets can be implemented. In general, an instruction set that temporally corresponds to a digital picture enables structuring knowledge of a device's operation at or around the time of the receiving or capturing the digital picture. Such functionality enables spontaneous or seamless learning of a device's operation in various visual surroundings as user operates the device in real life situations. In some designs, an instruction set can be received from a processor, application program, logic circuit, and/or other processing element as the instruction set is being used or executed. In other aspects, an instruction set can be received from a processor, application program, logic circuit, and/or other processing element after the instruction set is used or executed. In further aspects, an instruction set can be received from a processor, application program, logic circuit, and/or other processing element before the instruction set has been used or executed. An instruction set can be received from a running processor, running application program, running logic circuit, and/or other running processing element. As such, an instruction set can be received at runtime. In other designs, an instruction set can be received from an actuator. In some embodiments, an instruction set may include one or more commands, keywords, symbols (i.e. parentheses, brackets, commas, semicolons,

etc.), instructions, operators (i.e. =, <, >, etc.), variables, values, objects (i.e. file handle, network connection, Object1, etc.), data structures (i.e. table, database, user defined data structure, etc.), functions (i.e. Function1( ), FIRST( ), MIN( ), SQRT( ), etc.), parameters, states, signals, inputs, outputs, references thereto, and/or other components for performing an operation. In other embodiments, an instruction set may include source code, bytecode, intermediate code, compiled, interpreted, or otherwise translated code, runtime code, assembly code, machine code, and/or any other computer code. In further embodiments, an instruction set can be compiled, interpreted or otherwise translated into machine code or any intermediate code (i.e. bytecode, assembly code, etc.). In further embodiments, an instruction set may include one or more inputs into and/or outputs from a logic circuit. In further embodiments, an instruction set may include one or more inputs into an actuator. In some aspects, an instruction set can be received from memory (i.e. Memory **12**, etc.), hard drive, or any other storage element or repository. In other aspects, an instruction set can be received over a network such as Internet, local area network, wireless network, and/or other network. In further aspects, an instruction set can be received by an interface (i.e. Acquisition Interface **120**, etc.) configured to obtain instruction sets from a processor, application program, logic circuit, actuator, and/or other element. In general, an instruction set can be received by any element of the system. In some embodiments, receiving may be responsive to a triggering object, action, event, time, and/or other stimulus. Receiving comprises any action or operation by or for an Acquisition Interface **120**, Instruction Set **526**, and/or other disclosed elements.

At step **6115**, the first digital picture is correlated with the one or more instruction sets for operating the device. In some aspects, individual digital pictures can be correlated with one or more instruction sets. In other aspects, streams of digital pictures can be correlated with one or more instruction sets. In further aspects, individual digital pictures or streams of digital pictures can be correlated with temporally corresponding instruction sets as previously described. In further aspects, a digital picture or stream of digital pictures may not be correlated with any instruction sets. Correlating may include structuring or generating a knowledge cell (i.e. Knowledge Cell **800**, etc.) and storing one or more digital pictures correlated with any instruction sets into the knowledge cell. Therefore, knowledge cell may include any data structure or arrangement that can facilitate such storing. A knowledge cell includes a unit of knowledge of how a device operated in a visual surrounding. In some designs, extra information (i.e. Extra Info **527**, etc.) may optionally be used to facilitate enhanced comparisons or decision making in autonomous device operation where applicable. Therefore, any digital picture, instruction set, and/or other element may include or be correlated with extra information. Extra information may include any information useful in comparisons or decision making performed in autonomous device operation. Examples of extra information include time information, location information, computed information, observed information, sensory information, contextual information, and/or other information. In some embodiments, correlating may be responsive to a triggering object, action, event, time, and/or other stimulus. Correlating may be omitted where learning of a device's operations in visual surroundings is not implemented. Correlating comprises any action or operation by or for a Knowledge Structuring Unit **520**, Knowledge Cell **800**, and/or other disclosed elements.

At step **6120**, the first digital picture correlated with the one or more instruction sets for operating the device is stored. A digital picture correlated with one or more instruction sets may be part of a stored plurality of digital pictures correlated with one or more instruction sets. Digital pictures correlated with any instruction sets can be stored in a memory unit or other repository. The previously described knowledge cells comprising digital pictures correlated with any instruction sets can be used in/as neurons, nodes, vertices, or other elements in any of the data structures or arrangements (i.e. neural networks, graphs, sequences, collection of knowledge cells, etc.) used for storing the knowledge of a device's operation in visual surroundings. Knowledge cells may be connected, interrelated, or interlinked into knowledge structures using statistical, artificial intelligence, machine learning, and/or other models or techniques. Such interconnected or interrelated knowledge cells can be used for enabling autonomous device operation. The interconnected or interrelated knowledge cells may be stored or organized into a knowledgebase (i.e. Knowledgebase **530**, etc.). In some embodiments, knowledgebase may be or include a neural network (i.e. Neural Network **530***a*, etc.). In other embodiments, knowledgebase may be or include a graph (i.e. Graph **530***b*, etc.). In further embodiments, knowledgebase may be or include a collection of sequences (i.e. Collection of Sequences **530***c*, etc.). In further embodiments, knowledgebase may be or include a sequence (i.e. Sequence **533**, etc.). In further embodiments, knowledgebase may be or include a collection of knowledge cells (i.e. Collection of Knowledge Cells **530***d*, etc.). In general, knowledgebase may be or include any data structure or arrangement, and/or repository capable of storing the knowledge of a device's operation in various visual surroundings. Knowledgebase may also include or be substituted with various artificial intelligence methods, systems, and/or models for knowledge structuring, storing, and/or representation such as deep learning, supervised learning, unsupervised learning, neural networks (i.e. convolutional neural network, recurrent neural network, deep neural network, etc.), search-based, logic and/or fuzzy logic-based, optimization-based, tree/graph/other data structure-based, hierarchical, symbolic and/or sub-symbolic, evolutionary, genetic, multi-agent, deterministic, probabilistic, statistical, and/or other methods, systems, and/or models. Storing may be omitted where learning of a device's operations in visual surroundings is not implemented. Storing comprises any action or operation by or for a Knowledgebase **530**, Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, Knowledge Cell **800**, Node **852**, Layer **854**, Connection **853**, Similarity Comparison **125**, and/or other disclosed elements.

At step **6125**, a new digital picture is received. Step **6125** may include any action or operation described in Step **6105** as applicable.

At step **6130**, the new digital picture is compared with the first digital picture. Comparing one digital picture with another digital picture may include comparing at least a portion of one digital picture with at least a portion of the other digital picture. In some embodiments, digital pictures may be compared individually. In some aspects, comparing of individual pictures may include comparing one or more regions of one picture with one or more regions of another picture. In other aspects, comparing of individual pictures may include comparing one or more features of one picture with one or more features of another picture. In further aspects, comparing of individual pictures may include comparing pixels of one picture with pixels of another picture.

In other aspects, comparing of individual pictures may include recognizing a person or object in one digital picture and recognizing a person or object in another digital picture, and comparing the person or object from the one digital picture with the person or object from the other digital picture. Comparing may also include other aspects or properties of digital pictures or pixels examples of which comprise color adjustment, size adjustment, content manipulation, transparency (i.e. alpha channel, etc.), use of a mask, and/or others. In other embodiments, digital pictures may be compared collectively as part of streams of digital pictures (i.e. motion pictures, videos, etc.). In some aspects, collective comparing may include comparing one or more digital pictures of one stream of digital pictures with one or more digital pictures of another stream of digital pictures. In some aspects, Dynamic Time Warping (DTW) and/or other techniques can be utilized for comparison and/or aligning temporal sequences (i.e. streams of digital pictures, etc.) that may vary in time or speed. Any combination of the aforementioned and/or other elements or techniques can be utilized in alternate embodiments of the comparing. Comparing may be omitted where anticipating of a device's operation in visual surroundings is not implemented. Comparing comprises any action or operation by or for a Decision-making Unit **540**, Similarity Comparison **125**, and/or other disclosed elements.

At step **6135**, a determination is made that there is at least a partial match between the new digital picture and the first digital picture. In some embodiments, determining at least a partial match between individually compared digital pictures includes determining that similarity between one or more portions of one digital picture and one or more portions of another digital picture exceeds a similarity threshold. In other embodiments, determining at least a partial match between individually compared digital pictures includes determining at least a partial match between one or more portions of one digital picture and one or more portions of another digital picture. A portion of a digital picture may include a region, a feature, a pixel, or other portion. In further embodiments, determining at least a partial match between individually compared digital pictures includes determining that the number or percentage of matching or substantially matching regions of the compared pictures exceeds a threshold number (i.e. 1, 2, 5, 11, 39, etc.) or threshold percentage (i.e. 38%, 63%, 77%, 84%, 98%, etc.). In some aspects, the type of regions, the importance of regions, and/or other elements or techniques relating to regions can be utilized for determining similarity using regions. In further aspects, some of the regions can be omitted in determining similarity using regions. In further aspects, similarity determination can focus on regions of interest from the compared pictures. In further aspects, detection or recognition of persons or objects in regions of the compared pictures can be utilized for determining similarity. Where a reference to a region is used herein it should be understood that a portion of a region or a collection of regions can be used instead of or in addition to the region. In further embodiments, determining at least a partial match between individually compared digital pictures includes determining that the number or percentage of matching or substantially matching features of the compared pictures exceeds a threshold number (i.e. 3, 22, 47, 93, 128, 431, etc.) or a threshold percentage (i.e. 49%, 53%, 68%, 72%, 95%, etc.). In some aspects, the type of features, the importance of features, and/or other elements or techniques relating to features can be utilized for determining similarity using features. In further aspects, some of the features can be omitted in determining similarity using features. In further aspects, similarity determination can focus on features in certain regions of interest from the compared pictures. In further aspects, detection or recognition of persons or objects using features in the compared pictures can be utilized for determining similarity. Where a reference to a feature is used herein it should be understood that a portion of a feature or a collection of features can be used instead of or in addition to the feature. In further embodiments, determining at least a partial match between individually compared digital pictures may include determining that the number or percentage of matching or substantially matching pixels of the compared pictures exceeds a threshold number (i.e. 449, 2219, 92229, 442990, 1000028, etc.) or a threshold percentage (i.e. 39%, 45%, 58%, 72%, 92%, etc.). In some aspects, some of the pixels can be omitted in determining similarity using pixels. In further aspects, similarity determination can focus on pixels in certain regions of interest from the compared pictures. Where a reference to a pixel is used herein it should be understood that a collection of pixels can be used instead of or in addition to the pixel. In further embodiments, determining at least a partial match between individually compared digital pictures may include determining substantial similarity between at least a portion of one digital picture and at least a portion of another digital picture. In some aspects, substantial similarity of individually compared digital pictures can be achieved when a similarity between at least a portion of one digital picture and at least a portion of another digital picture exceeds a similarity threshold. In other aspects, substantial similarity of individually compared digital pictures can be achieved when the number or percentage of matching or substantially matching regions of the compared pictures exceeds a threshold number (i.e. 3, 22, 47, 93, 128, 431, etc.) or a threshold percentage (i.e. 49%, 53%, 68%, 72%, 95%, etc.). In further aspects, substantial similarity of individually compared digital pictures can be achieved when the number or percentage of matching or substantially matching features of the compared pictures exceeds a threshold number (i.e. 1, 2, 5, 11, 39, etc.) or threshold percentage (i.e. 38%, 63%, 77%, 84%, 98%, etc.). In further aspects, substantial similarity of individually compared digital pictures can be achieved when the number or percentage of matching or substantially matching pixels of the compared pictures exceeds a threshold number (i.e. 449, 2219, 92229, 442990, 1000028, etc.) or a threshold percentage (i.e. 39%, 45%, 58%, 72%, 92%, etc.). In some designs, substantial similarity of individually compared digital pictures can be achieved taking into account objects or persons detected within the compared digital pictures. For example, substantial similarity can be achieved if same or similar objects or persons are detected in the compared pictures. In some embodiments, determining at least a partial match between collectively compared digital pictures (i.e. streams of digital pictures [i.e. motion pictures, videos, etc.], etc.) may include determining that the number or percentage of matching or substantially matching digital pictures of the compared streams of digital pictures exceeds a threshold number (i.e. 28, 74, 283, 322, 995, 874, etc.) or a threshold percentage (i.e. 29%, 33%, 58%, 72%, 99%, etc.). In some aspects, Dynamic Time Warping (DTW) and/or other techniques for aligning temporal sequences (i.e. streams of digital pictures, etc.) that may vary in time or speed can be utilized in determining similarity of collectively compared digital pictures or streams digital pictures. In other aspects, the order of digital pictures, the importance of digital pictures, and/or other elements or techniques relating to digital pictures can be utilized for determining similarity of

collectively compared digital pictures or streams digital pictures. In further aspects, some of the digital pictures can be omitted in determining similarity of collectively compared digital pictures or streams digital pictures. In some designs, a threshold for a number or percentage similarity can be used to determine a match or substantial match between any of the aforementioned elements. Any combination of the aforementioned and/or other elements or techniques can be utilized in alternate embodiments. Determining may be omitted where anticipating of a device's operation in visual surroundings is not implemented. Determining comprises any action or operation by or for a Decision-making Unit **540**, Similarity Comparison **125**, and/or other disclosed elements.

At step **6140**, the one or more instruction sets for operating the device correlated with the first digital picture are executed. The executing may be performed in response to the aforementioned determining. The executing may be caused by VSADO Unit **100**, Artificial Intelligence Unit **110**, and/or other disclosed elements. An instruction set may be executed by a processor (i.e. Processor **11**, etc.), application program (i.e. Application Program **18**, etc.), logic circuit (i.e. Logic Circuit **250**, etc.), and/or other processing element. An instruction set may be executed or acted upon by an actuator. Executing may include executing one or more alternate instruction sets instead of or prior to an instruction set that would have been executed in a regular course of execution. In some aspects, alternate instruction sets comprise one or more instruction sets for operating a device correlated with one or more digital pictures. In some embodiments, executing may include modifying a register or other element of a processor with one or more alternate instruction sets. Executing may also include redirecting a processor to one or more alternate instruction sets. In other embodiments, processor may be or comprises a logic circuit. Executing may include modifying an element of a logic circuit with one or more alternate instruction sets, redirecting the logic circuit to one or more alternate instruction sets, replacing the inputs into the logic circuit with one or more alternate inputs or instruction sets, and/or replacing the outputs from the logic circuit with one or more alternate outputs or instruction sets. In further embodiments, a processor may include an application including instruction sets for operating a device, the application running on the processor. In some aspects, executing includes executing one or more alternate instruction sets as part of the application. In other aspects, executing includes modifying the application. In further aspects, executing includes redirecting the application to one or more alternate instruction sets. In further aspects, executing includes modifying one or more instruction sets of the application. In further aspects, executing includes modifying the application's source code, bytecode, intermediate code, compiled code, interpreted code, translated code, runtime code, assembly code, machine code, or other code. In further aspects, executing includes modifying memory, processor register, storage, repository or other element where the application's instruction sets are stored or used. In further aspects, executing includes modifying instruction sets used for operating an object of the application. In further aspects, executing includes modifying an element of a processor, an element of a device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input used in running the application. In further aspects, executing includes modifying the application at source code write time, compile time, interpretation time, translation time, linking time, loading time, runtime, or other time. In further aspects, executing includes modifying one or more of the application's lines of code, statements, instructions, functions, routines, subroutines, basic blocks, or other code segments. In further aspects, executing includes a manual, automatic, dynamic, just in time (JIT), or other instrumentation of the application. In further aspects, executing includes utilizing one or more of a .NET tool, .NET application programming interface (API), Java tool, Java API, operating system tool, independent tool or other tool for modifying the application. In further aspects, executing includes utilizing a dynamic, interpreted, scripting or other programming language. In further aspects, executing includes utilizing dynamic code, dynamic class loading, or reflection. In further aspects, executing includes utilizing assembly language. In further aspects, executing includes utilizing metaprogramming, self-modifying code, or an application modification tool. In further aspects, executing includes utilizing just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further aspects, executing includes utilizing dynamic expression creation, dynamic expression execution, dynamic function creation, or dynamic function execution. In further aspects, executing includes adding or inserting additional code into the application's code. In further aspects, executing includes modifying, removing, rewriting, or overwriting the application's code. In further aspects, executing includes branching, redirecting, extending, or hot swapping the application's code. Branching or redirecting an application's code may include inserting a branch, jump, or other means for redirecting the application's execution. Executing comprises any action or operation by or for a Processor **11**, Application Program **18**, Logic Circuit **250**, Modification Interface **130**, and/or other disclosed elements.

At step **6145**, one or more operations defined by the one or more instruction sets for operating the device correlated with the first digital picture are performed by the device. The one or more operations may be performed in response to the aforementioned executing. An operation includes any operation that can be performed by, with, or on the device. An operation includes any operation that can be performed by, with, or on an actuator. In one example, an operation includes any operation (i.e. moving, maneuvering, collecting, unloading, lifting, screwing, gripping, etc.) with or by a computing enabled machine (i.e. Computing Enabled Machine **98***a*, etc.). In a further example, an operation includes any operation with or by a fixture (i.e. Fixture **98***b*, etc.). In a further example, an operation includes any operation (i.e. setting, starting, stopping, etc.) on or by a control device (i.e. Control Device **98***c*, etc.). In one example, an operation includes any operation on a smartphone (i.e. Smartphone **98***d*, etc.) or other mobile computer. In a further example, an operation includes any operation on or by a computer or computing enabled device. In a further example, an operation includes any motion or operation on or by an actuator. One of ordinary skill in art will recognize that, while all possible variations of operations on a device are too voluminous to list and limited only by the device's design and/or user's utilization, other operations are within the scope of this disclosure in various implementations.

Referring to FIG. **32**, the illustration shows an embodiment of a method **6200** for learning and/or using visual surrounding for autonomous device operation. The method can be used on a computing device or system to enable learning of a device's operation in various visual surroundings and enable autonomous device operation in similar visual surroundings. Method **6200** may include any action or operation of any of the disclosed methods such as method

6100, **6300**, **6400**, **6500**, **6600**, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method **6200**.

At step **6205**, a first digital picture is received. Step **6205** may include any action or operation described in Step **6105** of method **6100** as applicable.

At step **6210**, one or more instruction sets for operating a device are received. Step **6210** may include any action or operation described in Step **6110** of method **6100** as applicable.

At step **6215**, the first digital picture correlated with the one or more instruction sets for operating the device are learned. Step **6215** may include any action or operation described in Step **6115** and/or Step **6120** of method **6100** as applicable.

At step **6220**, a new digital picture is received. Step **6220** may include any action or operation described in Step **6125** of method **6100** as applicable.

At step **6225**, the one or more instruction sets for operating the device correlated with the first digital picture are anticipated based on at least a partial match between the new digital picture and the first digital picture. Step **6225** may include any action or operation described in Step **6130** and/or Step **6135** of method **6100** as applicable.

At step **6230**, the one or more instruction sets for operating the device correlated with the first digital picture are executed. Step **6230** may include any action or operation described in Step **6140** of method **6100** as applicable.

At step **6235**, one or more operations defined by the one or more instruction sets for operating the device correlated with the first digital picture are performed by the device. Step **6235** may include any action or operation described in Step **6145** of method **6100** as applicable.

Referring to FIG. **33**, the illustration shows an embodiment of a method **6300** for learning and/or using visual surrounding for autonomous device operation. The method can be used on a computing device or system to enable learning of a device's operation in various visual surroundings and enable autonomous device operation in similar visual surroundings. Method **6300** may include any action or operation of any of the disclosed methods such as method **6100**, **6200**, **6400**, **6500**, **6600**, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method **6300**.

At step **6305**, a first stream of digital pictures is received. Step **6305** may include any action or operation described in Step **6105** of method **6100** as applicable.

At step **6310**, one or more instruction sets for operating a device are received. Step **6310** may include any action or operation described in Step **6110** of method **6100** as applicable.

At step **6315**, the first stream of digital pictures is correlated with the one or more instruction sets for operating the device. Step **6315** may include any action or operation described in Step **6115** of method **6100** as applicable.

At step **6320**, the first stream of digital pictures correlated with the one or more instruction sets for operating the device is stored. Step **6320** may include any action or operation described in Step **6120** of method **6100** as applicable.

At step **6325**, a new stream of digital pictures is received. Step **6325** may include any action or operation described in Step **6125** of method **6100** as applicable.

At step **6330**, the new stream of digital pictures is compared with the first stream of digital pictures. Step **6330** may include any action or operation described in Step **6130** of method **6100** as applicable.

At step **6335**, a determination is made that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures. Step **6335** may include any action or operation described in Step **6135** of method **6100** as applicable.

At step **6340**, the one or more instruction sets for operating the device correlated with the first stream of digital pictures are executed. Step **6340** may include any action or operation described in Step **6140** of method **6100** as applicable.

At step **6345**, one or more operations defined by the one or more instruction sets for operating the device correlated with the first stream of digital pictures are performed by the device. Step **6345** may include any action or operation described in Step **6145** of method **6100** as applicable.

Referring to FIG. **34**, the illustration shows an embodiment of a method **6400** for learning and/or using visual surrounding for autonomous device operation. The method can be used on a computing device or system to enable learning of a device's operation in various visual surroundings and enable autonomous device operation in similar visual surroundings. Method **6400** may include any action or operation of any of the disclosed methods such as method **6100**, **6200**, **6300**, **6500**, **6600**, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method **6400**.

At step **6405**, a first digital picture is received. Step **6405** may include any action or operation described in Step **6105** of method **6100** as applicable.

At step **6410**, at least one input are received, wherein the at least one input are also received by a logic circuit, and wherein the logic circuit is configured to receive inputs and produce outputs, the outputs for operating a device. Step **6410** may include any action or operation described in Step **6110** of method **6100** as applicable.

At step **6415**, the first digital picture is correlated with the at least one input. Step **6415** may include any action or operation described in Step **6115** of method **6100** as applicable.

At step **6420**, the first digital picture correlated with the at least one input is stored. Step **6420** may include any action or operation described in Step **6120** of method **6100** as applicable.

At step **6425**, a new digital picture is received. Step **6425** may include any action or operation described in Step **6125** of method **6100** as applicable.

At step **6430**, the new digital picture is compared with the first digital picture. Step **6430** may include any action or operation described in Step **6130** of method **6100** as applicable.

At step **6435**, a determination is made that there is at least a partial match between the new digital picture and the first digital picture. Step **6435** may include any action or operation described in Step **6135** of method **6100** as applicable.

At step **6440**, the at least one input correlated with the first digital picture are received by the logic circuit.

Step **6440** may include any action or operation described in Step **6140** of method **6100** as applicable.

At step **6445**, at least one operation defined by at least one output for operating the device produced by the logic circuit

are performed by the device. Step **6445** may include any action or operation described in Step **6145** of method **6100** as applicable.

Referring to FIG. **35**, the illustration shows an embodiment of a method **6500** for learning and/or using visual surrounding for autonomous device operation. The method can be used on a computing device or system to enable learning of a device's operation in various visual surroundings and enable autonomous device operation in similar visual surroundings. Method **6500** may include any action or operation of any of the disclosed methods such as method **6100**, **6200**, **6300**, **6400**, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method **6500**.

At step **6505**, a first digital picture is received. Step **6505** may include any action or operation described in Step **6105** of method **6100** as applicable.

At step **6510**, at least one output are received, the at least one output transmitted from a logic circuit, wherein the logic circuit is configured to receive inputs and produce outputs, the outputs for operating a device. Step **6510** may include any action or operation described in Step **6110** of method **6100** as applicable.

At step **6515**, the first digital picture is correlated with the at least one output. Step **6515** may include any action or operation described in Step **6115** of method **6100** as applicable.

At step **6520**, the first digital picture correlated with the at least one output is stored. Step **6520** may include any action or operation described in Step **6120** of method **6100** as applicable.

At step **6525**, a new digital picture is received. Step **6525** may include any action or operation described in Step **6125** of method **6100** as applicable.

At step **6530**, the new digital picture is compared with the first digital picture. Step **6530** may include any action or operation described in Step **6130** of method **6100** as applicable.

At step **6535**, a determination is made that there is at least a partial match between the new digital picture and the first digital picture. Step **6535** may include any action or operation described in Step **6135** of method **6100** as applicable.

At step **6540**, at least one operation defined by the at least one output correlated with the first digital picture are performed by the device. Step **6540** may include any action or operation described in Step **6145** of method **6100** as applicable.

Referring to FIG. **36**, the illustration shows an embodiment of a method **6600** for learning and/or using visual surrounding for autonomous device operation. The method can be used on a computing device or system to enable learning of a device's operation in various visual surroundings and enable autonomous device operation in similar visual surroundings. Method **6600** may include any action or operation of any of the disclosed methods such as method **6100**, **6200**, **6300**, **6400**, **6500**, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method **6600**.

At step **6605**, a first digital picture is received. Step **6605** may include any action or operation described in Step **6105** of method **6100** as applicable.

At step **6610**, at least one input are received, wherein the at least one input are also received by an actuator, and wherein the actuator is configured to receive inputs and perform motions. Step **6610** may include any action or operation described in Step **6110** of method **6100** as applicable.

At step **6615**, the first digital picture is correlated with the at least one input. Step **6615** may include any action or operation described in Step **6115** of method **6100** as applicable.

At step **6620**, the first digital picture correlated with the at least one input is stored. Step **6620** may include any action or operation described in Step **6120** of method **6100** as applicable.

At step **6625**, a new digital picture is received. Step **6625** may include any action or operation described in Step **6125** of method **6100** as applicable.

At step **6630**, the new digital picture is compared with the first digital picture. Step **6630** may include any action or operation described in Step **6130** of method **6100** as applicable.

At step **6635**, a determination is made that there is at least a partial match between the new digital picture and the first digital picture. Step **6635** may include any action or operation described in Step **6135** of method **6100** as applicable.

At step **6640**, the at least one input correlated with the first digital picture are received by the actuator. Step **6640** may include any action or operation described in Step **6140** of method **6100** as applicable.

At step **6645**, at least one motion defined by the at least one input correlated with the first digital picture are performed by the actuator. Step **6645** may include any action or operation described in Step **6145** of method **6100** as applicable.

Referring to FIG. **37**, in some exemplary embodiments, Device **98** may be or include a Computing-enabled Machine **98***a*. Examples of Computing-enabled Machine **98***a* comprise a loader, a bulldozer, an excavator, a crane, a forklift, a truck, an assembly machine, a material/object handling machine, a sorting machine, an industrial machine, a kitchen appliance, a robot, a tank, an airplane, a helicopter, a vessel, a submarine, a ground/aerial/aquatic vehicle, and/or other computing-enabled machine. In some aspects, Computing-enabled Machine **98***a* may itself include computing capabilities. In other aspects, computing capabilities may be included in a remote computing device (i.e. server, etc.) and provided to Computing-enabled Machine **98***a* (i.e. via a network, etc.). Computing-enabled Machine **98***a* may be operated by User **50** in person or remotely. Computing-enabled Machine **98***a* may include Picture Capturing Apparatus **90** such as a motion picture, still picture, or other camera that captures one or more Digital Pictures **525** of Computing-enabled Machine's **98***d* surrounding. Computing-enabled Machine **98***a* may also include or be controlled by Logic Circuit **250** (i.e. microcontroller, etc.), Processor **11** (i.e. including any Application Program **18** running thereon, etc.), and/or other processing element that receives User's **50** (i.e. operator's, etc.) operating directions and causes desired operations with Computing-enabled Machine **98***a* such as moving, maneuvering, collecting, unloading, pushing, digging, lifting, and/or other operations. User **50** can interact with Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element through inputting operating directions (i.e. manipulating levers, pressing buttons, etc.) via Human-machine Interface **23** such as one or more levers or other input device. For instance, responsive to User's **50** manipulating one or more levers, Logic Circuit **250** or Processor **11** may cause Computing-enabled Machine's **98***d* arm with bucket to collect a load,

one or more motors or other actuators to move or maneuver Computing-enabled Machine **98***a*, lifting system (i.e. hydraulic, pneumatic, mechanical, electrical, etc.) to lift a load, and/or arm with bucket to unload a load. Computing-enabled Machine **98***a* may also include or be coupled to VSADO Unit **100**. VSADO Unit **100** may be embedded (i.e. integrated, etc.) into or coupled to Computing-enabled Machine's **98***d* Logic Circuit **250**, Processor **11**, and/or other processing element. VSADO Unit **100** may also be a program embedded (i.e. integrated, etc.) into or interfaced with Application Program **18** running on Processor **11** and/or other processing element. VSADO Unit **100** can obtain Instruction Sets **526** used or executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element. In some aspects, Instruction Sets **526** may include one or more inputs into or outputs from Computing-enabled Machine's **98***d* Logic Circuit **250** (i.e. microcontroller, etc.). In other aspects, Instruction Sets **526** may include one or more instruction sets from Computing-enabled Machine's **98***d* Processor's **11** registers or other components. In further aspects, Instruction Sets **526** may include one or more instruction sets used or executed in Application Program **18** running on Processor **11** and/or other processing element. VSADO Unit **100** may also optionally obtain any Extra Info **527** (i.e. time, location, computed, observed, sensory, and/or other information, etc.) related to Computing-enabled Machine's **98***d* operation. As User **50** operates Computing-enabled Machine **98***a* in various visual surroundings as shown, VSADO Unit **100** may learn Computing-enabled Machine's **98***d* operations in visual surroundings by correlating Digital Pictures **525** of Computing-enabled Machine's **98***d* surrounding with one or more Instruction Sets **526** used or executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element. Any Extra Info **527** related to Computing-enabled Machine's **98***d* operation may also optionally be correlated with Digital Pictures **525** of Computing-enabled Machine's **98***d* surrounding. VSADO Unit **100** may store this knowledge into Knowledgebase **530** (i.e. Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.). In the future, VSADO Unit **110** may compare incoming Digital Pictures **525** of Computing-enabled Machine's **98***d* surrounding with previously learned Digital Pictures **525** including optionally using any Extra Info **527** for enhanced decision making. If substantially similar or at least a partial match is found or determined, the Instruction Sets **526** correlated with the previously learned Digital Pictures **525** can be autonomously executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element, thereby enabling autonomous operation of Computing-enabled Machine **98***a* in a similar visual surrounding as in a previously learned one. For instance, Computing-enabled Machine **98***a* (i.e. loader, etc.) comprising VSADO Unit **100** may learn User **50**-directed collecting, moving, maneuvering, lifting, and/or unloading in a visual surrounding that includes a pile of material, truck, and/or other objects with which Computing-enabled Machine **98***a* may need to interact. In the future, when visual surrounding that includes same or similar objects is encountered, or when same or similar objects are detected, Computing-enabled Machine **98***a* may implement collecting, moving, maneuvering, lifting, and/or unloading autonomously.

Referring to FIG. **38**, in some exemplary embodiments, Device **98** may be or include a Computing-enabled Machine **98***a* comprising or coupled to a plurality of Picture Capturing Apparatuses **90**. In one example, different Picture Capturing Apparatuses **90** may capture Digital Pictures **525** of different angles of Computing-enabled Machine's **98***d* front. In another example, different Picture Capturing Apparatuses **90** may capture Digital Pictures **525** of the front, sides, and/or back of Computing-enabled Machine **98***a*. In a further example as shown, different Picture Capturing Apparatuses **90** may be placed on different sub-devices, sub-systems, or elements of Computing-enabled Machine **98***a*. Specifically, for instance, Picture Capturing Apparatus **90***a* may be placed on the roof of Computing-enabled Machine **98***a* (i.e. loader, etc.), Picture Capturing Apparatus **90***b* may be placed on the arm of Computing-enabled Machine **98***a*, and Picture Capturing Apparatus **90***c* may be placed on the bucket of Computing-enabled Machine **98***a*. In some designs where multiple Picture Capturing Apparatuses **90** are utilized, as User **50** operates Computing-enabled Machine **98***a* in various visual surroundings, VSADO Unit **100** may learn Computing-enabled Machine's **98***d* operations in visual surroundings by correlating collective Digital Pictures **525** of Computing-enabled Machine's **98***d* surrounding from multiple Picture Capturing Apparatuses **90** with one or more Instruction Sets **526** used or executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element. In other designs where multiple Picture Capturing Apparatuses **90** are utilized, multiple VSADO Units **100** may also be utilized (i.e. one VSADO Unit **100** for each Picture Capturing Apparatus **90**, etc.). In such designs, as User **50** operates Computing-enabled Machine **98***a* in various visual surroundings, VSADO Unit **100** may learn Computing-enabled Machine's **98***d* operations in visual surroundings by correlating Digital Pictures **525** of Computing-enabled Machine's **98***d* surrounding from Picture Capturing Apparatus **90** assigned to the VSADO Unit **100** with one or more Instruction Sets **526** used or executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element. Each sub-device, sub-system, or element can, therefore, perform its own learning and/or decision making in autonomous operation.

In some embodiments, Computing-enabled Machine **98***a* may include a plurality of Logic Circuits **250** (i.e. microcontrollers, etc.), Processors **11**, Application Programs **18**, and/or other processing elements. In some aspects, each processing element may control a sub-device, sub-system, or element of Computing-enabled Machine's **98***d*. For example, one Processor **11** (i.e. including any Application Programs **18** running thereon, etc.) may control the moving system (i.e. drivetrain, powertrain, etc.) of Computing-enabled Machine **98***a* (i.e. loader), one Logic Circuit **250** may control an arm of Computing-enabled Machine **98***a*, and a second Logic Circuit **250** may control a bucket of Computing-enabled Machine **98***a*. In some designs where multiple processing elements are utilized, as User **50** operates Computing-enabled Machine **98***a* in various visual surroundings, VSADO Unit **100** may learn Computing-enabled Machine's **98***d* operations in visual surroundings by correlating Digital Pictures **525** of Computing-enabled Machine's **98***d* surrounding with collective one or more Instruction Sets **526** used or executed by a plurality of Logic Circuits **250**, Processors **11**, Application Programs **18**, and/or other processing elements. In other designs where multiple processing elements are utilized, multiple VSADO Units **100** may also be utilized (i.e. one VSADO Unit **100** for each processing element, etc.). In such designs, as User **50** operates Computing-enabled Machine **98***a* in various visual surroundings, VSADO Unit **100** may learn Computing-enabled Machine's **98***d* operations in visual surround-

ings by correlating Digital Pictures **525** of Computing-enabled Machine's **98***d* surrounding with one or more Instruction Sets **526** used or executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element assigned to the VSADO Unit **100**.

In some embodiments, Computing-enabled Machine **98***a* (i.e. loader, etc.) may be controlled by a combination of VSADO Unit **100** and other systems and/or techniques. In some aspects, Computing-enabled Machine **98***a* controlled by VSADO Unit **100** may encounter a visual surrounding that has not been encountered or learned before. In such situations, User **50** and/or non-VSADO system may take control of Computing-enabled Machine's **98***d* operation. VSADO Unit **100** may take control again when Computing-enabled Machine **98***a* encounters a previously learned visual surrounding. Naturally, VSADO Unit **100** can learn Computing-enabled Machine's **98***d* operation in visual surroundings while User **50** and/or non-VSADO system is in control of Device **98**, thereby reducing or eliminating the need for future involvement of User **50** and/or non-VSADO system. In some implementations, one User **50** can control or assist in controlling multiple Computing-enabled Machines **98***d* comprising VSADO Units **100**. For example, User **50** can control or assist in controlling a Computing-enabled Machine **98***a* that may encounter a visual surrounding that has not been encountered or learned before while the Computing-enabled Machines **98***d* operating in previously learned visual surroundings can operate autonomously. In other aspects, Computing-enabled Machine **98***a* may be primarily controlled by User **50** and/or non-VSADO system. User **50** and/or non-VSADO system can release control to VSADO Unit **100** for any reason (i.e. User **50** gets tired or distracted, non-VSADO system gets stuck or cannot make a decision, etc.), at which point Computing-enabled Machine **98***a* can be controlled by VSADO Unit **100**. In further aspects, VSADO Unit **100** may take control in certain special visual surroundings where VSADO Unit **100** may offer superior performance even if User **50** and/or non-VSADO system may generally be preferred. Once Computing-enabled Machine **98***a* leaves such special visual surrounding, VSADO Unit **100** may release control to User **50** and/or non-VSADO system. In general, VSADO Unit **100** can take control from, share control with, or release control to User **50**, non-VSADO system, and/or other system or process at any time, under any circumstances, and remain in control for any period of time as needed.

In some embodiments, VSADO Unit **100** may control one or more sub-devices, sub-systems, or elements of Computing-enabled Machine **98***a* (i.e. loader) while User **50** and/or non-VSADO system may control other one or more sub-devices, sub-systems, or elements of Computing-enabled Machine **98***a*. For example, User **50** and/or non-VSADO system may control the moving system (i.e. drivetrain, powertrain, etc.) of Computing-enabled Machine **98***a*, while VSADO Unit **100** may control an arm and bucket of Computing-enabled Machine **98***a*. Any other combination of controlling various sub-devices, sub-systems, or elements of Computing-enabled Machine **98***a* by VSADO Unit **100** and User **50** and/or non-VSADO system can be implemented.

One of ordinary skill in art will understand that the features, functionalities, and embodiments described with respect to Computing-enabled Machine **98***a* can similarly be implemented on any computing enabled machine such as a bulldozer, an excavator, a crane, a forklift, a truck, an assembly machine, a material/object handling machine, a sorting machine, an industrial machine, a kitchen appliance, a robot, a tank, an airplane, a helicopter, a vessel, a submarine, a ground/aerial/aquatic vehicle, and/or other computing-enabled machine.

Referring to FIG. **39**, in some exemplary embodiments, Device **98** may be or include a Fixture **98***b*. Examples of Fixture **98***b* comprise a fan, a light, automated blind, and/or other fixture. Fixture **98***b* may include Picture Capturing Apparatus **90** such as a motion picture, still picture, or other camera that captures one or more Digital Pictures **525** of Fixture's **98***b* surrounding. Fixture **98***b* may also include or be controlled by Logic Circuit **250** (i.e. microcontroller, etc.), Processor **11** (i.e. including any Application Program **18** running thereon, etc.), and/or other processing element that receives User's **50** operating directions and causes desired operations with Fixture **98***b* such as setting speed of a fan, adjusting intensity of a light, adjusting angle of an automated blind, and/or other operations. User **50** can interact with Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element through inputting operating directions (i.e. pressing control buttons, switching switches, etc.) via Human-machine Interface **23** such as a controller, switch, or other input device. For instance, responsive to User's **50** pressing a control button, Logic Circuit **250** or Processor **11** may cause Fixture **98***b* to set a speed (i.e. in the case of a fan, etc.). Fixture **98***b* may also include or be coupled to VSADO Unit **100**. VSADO Unit **100** may be embedded (i.e. integrated, etc.) into or coupled to Fixture's **98***b* Logic Circuit **250**, Processor **11**, and/or other processing element. VSADO Unit **100** may also be a program embedded (i.e. integrated, etc.) into or interfaced with Application Program **18** running on Processor **11** and/or other processing element. VSADO Unit **100** can obtain Instruction Sets **526** used or executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element. In some aspects, Instruction Sets **526** may include one or more inputs into or outputs from Fixture's **98***b* Logic Circuit **250** (i.e. microcontroller, etc.). In other aspects, Instruction Sets **526** may include one or more instruction sets from Fixture's **98***b* Processor's **11** registers or other components. In further aspects, Instruction Sets **526** may include one or more instruction sets used or executed in Application Program **18** running on Processor **11** and/or other processing element. VSADO Unit **100** may also optionally obtain any Extra Info **527** (i.e. time, location, computed, observed, sensory, and/or other information, etc.) related to Fixture's **98***b* operation. As User **50** operates Fixture **98***b* in a visual surrounding as shown, VSADO Unit **100** may learn Fixture's **98***b* operation in the visual surrounding by correlating Digital Pictures **525** of Fixture's **98***b* surrounding with one or more Instruction Sets **526** used or executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element. Any Extra Info **527** related to Fixture's **98***b* operation may also optionally be correlated with Digital Pictures **525** of Fixture's **98***b* surrounding. VSADO Unit **100** may store this knowledge into Knowledgebase **530** (i.e. Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.). In the future, VSADO Unit **110** may compare incoming Digital Pictures **525** of Fixture's **98***b* surrounding with previously learned Digital Pictures **525** including optionally using any Extra Info **527** for enhanced decision making. If substantially similar or at least a partial match is found or determined, the Instruction Sets **526** correlated with the previously learned Digital Pictures **525** can be autonomously executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element, thereby enabling autonomous operation

of Fixture **98***b* in a similar visual surrounding as in a previously learned one. For instance, Fixture **98***b* (i.e. ceiling fan, etc.) comprising VSADO Unit **100** may learn User's **50** setting speed of Fixture **98***b* in a visual surrounding that includes User **50** entering or being present in a room. In the future, when visual surrounding that includes User **50** entering or being present in the room, or when User **50** or his/her body part (i.e. face, etc.) is detected, Fixture **98***b* may implement setting of its speed autonomously. In some aspects, Fixture **98***b* comprising VSADO Unit **100** may engage autonomous operation (i.e. autonomous fan speed setting, etc.) if a specific person is detected by using facial recognition, thereby personalizing the operation of Fixture **98***b*. In other aspects, Fixture **98***b* may engage autonomous operation (i.e. autonomous fan speed setting, etc.) if any person is detected by using person or object recognition.

Referring to FIG. **40**, in some exemplary embodiments, Device **98** may be or include a Control Device **98***c*. Examples of Control Device **98***c* comprise a thermostat, a control panel, a remote or other controller, and/or other control device. Control Device **98***c* may include Picture Capturing Apparatus **90** such as a motion picture, still picture, or other camera that captures one or more Digital Pictures **525** of Control Device's **98***c* surrounding. Control Device **98***c* may also include Logic Circuit **250** (i.e. microcontroller, etc.), Processor **11** (i.e. including any Application Program **18** running thereon, etc.), and/or other processing element that receives User's **50** operating directions and causes desired operations on a device or system controlled by Control Device **98***c* such as regulating temperature of an air conditioning system, and/or other operations. User **50** can interact with Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element through inputting operating directions (i.e. pressing control buttons, etc.) via Human-machine Interface **23** such as a control panel or other input device. For instance, responsive to User's **50** pressing a control button, Logic Circuit **250** or Processor **11** may cause Control Device **98***c* to increase or decrease a temperature of an air conditioning system. Control Device **98***c* may also include or be coupled to VSADO Unit **100**. VSADO Unit **100** may be embedded (i.e. integrated, etc.) into or coupled to Control Device's **98***c* Logic Circuit **250**, Processor **11**, and/or other processing element. VSADO Unit **100** may also be a program embedded (i.e. integrated, etc.) into or interfaced with Application Program **18** running on Processor **11** and/or other processing element. VSADO Unit **100** can obtain Instruction Sets **526** used or executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element. In some aspects, Instruction Sets **526** may include one or more inputs into or outputs from Control Device's **98***c* Logic Circuit **250** (i.e. microcontroller, etc.). In other aspects, Instruction Sets **526** may include one or more instruction sets from Control Device's **98***c* Processor's **11** registers or other components. In further aspects, Instruction Sets **526** may include one or more instruction sets used or executed in Application Program **18** running on Processor **11** and/or other processing element. VSADO Unit **100** may also optionally obtain any Extra Info **527** (i.e. time, location, computed, observed, sensory, and/or other information, etc.) related to Control Device's **98***c* operation. As User **50** operates Control Device **98***c* in a visual surrounding as shown, VSADO Unit **100** may learn Control Device's **98***c* operation in the visual surrounding by correlating Digital Pictures **525** of Control Device's **98***c* surrounding with one or more Instruction Sets **526** used or executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element. Any Extra Info **527** related to Control Device's **98***c* operation may also optionally be correlated with Digital Pictures **525** of Control Device's **98***c* surrounding. VSADO Unit **100** may store this knowledge into Knowledgebase **530** (i.e. Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.). In the future, VSADO Unit **110** may compare incoming Digital Pictures **525** of Control Device's **98***c* surrounding with previously learned Digital Pictures **525** including optionally using any Extra Info **527** for enhanced decision making. If substantially similar or at least a partial match is found or determined, the Instruction Sets **526** correlated with the previously learned Digital Pictures **525** can be autonomously executed by Logic Circuit **250**, Processor **11**, Application Program **18**, and/or other processing element, thereby enabling autonomous operation of Control Device **98***c* in a similar visual surrounding as in a previously learned one. For instance, Control Device **98***c* comprising VSADO Unit **100** may learn User's **50** setting temperature of an air conditioning system controlled by Control Device **98***c* in a visual surrounding that includes User **50** entering or being present in a room. In the future, when visual surrounding that includes User **50** entering or being present in the room, or when User **50** or his/her body part (i.e. face, etc.) is detected, Control Device **98***c* may implement setting temperature of the air conditioning system autonomously. In some aspects, Control Device **98***c* may engage autonomous operation (i.e. autonomous temperature setting of an air conditioning system, etc.) if a specific person is detected by using facial recognition, thereby personalizing the operation of Control Device **98***c*. In other aspects, Control Device **98***c* may engage autonomous operation (i.e. autonomous temperature setting of an air conditioning system, etc.) if any person is detected by using person or object recognition. Referring to FIG. **41**, in some exemplary embodiments, Device **98** may be or include a Smartphone **98***d*. Examples of Smartphone **98***d* comprise Apple iPhone, Samsung Galaxy, Microsoft Lumia, and/or other smartphone. Smartphone **98***d* may include Picture Capturing Apparatus **90** such as a motion picture, still picture, or other camera that captures one or more Digital Pictures **525** of Smartphone's **98***a* surrounding. Smartphone **98***d* may include Processor **11** and one or more Application Programs **18** such as a phone control application that receives User's **50** operating directions and causes desired operations with Smartphone **98***d* such as making a call, ending a call, increasing volume, setting Smartphone **98***d* on vibrate mode, and/or other operations. User **50** can interact with Processor **11** and/or Application Program **18** through inputting operating directions (i.e. touching touchscreen elements, etc.) via Human-machine Interface **23** such as a touchscreen or other input device. For instance, responsive to User's **50** touching a touchscreen element, Processor **11** and/or Application Program **18** may cause

Smartphone **98***d* to go into a vibrate mode. Smartphone **98***d* may also include or be coupled to VSADO Unit **100**. VSADO Unit **100** may be embedded (i.e. integrated, etc.) into or coupled to Smartphone's **98***a* Processor **11** and/or other processing element. VSADO Unit **100** may also be a program embedded (i.e. integrated, etc.) into or interfaced with Application Program **18** running on Processor **11** and/or other processing element. VSADO Unit **100** can obtain Instruction Sets **526** used or executed by Processor **11**, Application Program **18**, and/or other processing element. In some aspects, Instruction Sets **526** may include one or more instruction sets used or executed in Application Program **18** running on Processor **11** and/or other processing element. In

other aspects, Instruction Sets **526** may include one or more instruction sets from Smartphone's **98***a* Processor's **11** registers or other components. VSADO Unit **100** may also optionally obtain any Extra Info **527** (i.e. time, location, computed, observed, sensory, and/or other information, etc.) related to Smartphone's **98***a* operation. As User **50** operates Smartphone **98***d* in a visual surroundings as shown, VSADO Unit **100** may learn Smartphone's **98***a* operation in the visual surrounding by correlating Digital Pictures **525** of Smartphone's **98***a* surrounding with one or more Instruction Sets **526** used or executed by Processor **11**, Application Program **18**, and/or other processing element. Any Extra Info **527** related to Smartphone's **98***a* operation may also optionally be correlated with Digital Pictures **525** of Smartphone's **98***a* surrounding. VSADO Unit **100** may store this knowledge into Knowledgebase **530** (i.e. Neural Network **530***a*, Graph **530***b*, Collection of Sequences **530***c*, Sequence **533**, Collection of Knowledge Cells **530***d*, etc.). In the future, VSADO Unit **110** may compare incoming Digital Pictures **525** of Smartphone's **98***a* surrounding with previously learned Digital Pictures **525** including optionally using any Extra Info **527** for enhanced decision making. If substantially similar or at least a partial match is found or determined, the Instruction Sets **526** correlated with the previously learned Digital Pictures **525** can be autonomously executed by Processor **11**, Application Program **18**, and/or other processing element, thereby enabling autonomous operation of Smartphone **98***d* in a similar visual surrounding as in a previously learned one. For instance, Smartphone **98***d* comprising VSADO Unit **100** may learn User's **50** setting of Smartphone **98***d* on vibrate mode in a visual surrounding that includes a classroom. In the future, when visual surrounding that includes a classroom is encountered, or when classroom is detected, Smartphone **98***d* may implement vibrate setting autonomously. In some aspects, similar functionality can be utilized in visual surroundings that include a house of worship, cemetery, and/or others.

In some embodiments, VSADO Unit **100** can be used to enable Smartphone **98***d*, computer, and/or application to learn User's **50** movements for interacting with or controlling Smartphone **98***d*, computer, and/or application. In one example, while viewing a web page in a web browser running on Smartphone **98***d*, User **50** may perform a head nod during or after which User **50** may scroll down the web page. Smartphone **98***d* comprising VSADO Unit **100** may learn User's **50** scrolling of a web page in a visual surrounding that includes User **50** performing a head nod. In the future, when visual surrounding that includes User **50** performing a head nod is encountered or detected, Smartphone **98***d* may implement scrolling of a web page in a web browser autonomously. In another example, while operating a user controllable object (i.e. avatar, etc.) in a computer game running on Smartphone **98***d*, User **50** may lean right during or after which User **50** may direct the user controllable object to turn or steer right. Smartphone **98***d* comprising VSADO Unit **100** may learn User's **50** directing the user controllable object to turn or steer right in a visual surrounding that includes User **50** leaning right. In the future, when visual surrounding that includes User **50** leaning right is encountered or detected, Smartphone **98***d* may implement directing the user controllable object to turn or steer right in a computer game autonomously. Therefore, VSADO Unit **100** can spontaneously learn both User's **50** movements and Instruction Sets **526** implementing an operation without User **50** needing to program, manually designate, or otherwise assign the movements to Instruction Sets **526** implementing the operation. Such functionality enables learning of User **50**-chosen movements and User **50**-chosen operations seamlessly as User **50** operates a device, application, and/or object thereof in real life situations without the need for special training sessions. Any User's **50** movements can be utilized examples of which include moving head, moving facial parts (i.e. eyes, lips, etc.), moving shoulders, moving hands, moving hand parts (i.e. fingers, etc.), moving body, moving body parts (i.e. arms, legs, etc.), and/or others. Any of the functionalities described with respect to Smartphone **98***d* similarly apply to any computer or computing enabled device.

It must be noted that as used herein and in the appended claims, the singular forms "a", "an", and "the" include plural referents unless the context clearly dictates otherwise.

A number of embodiments have been described herein. While this disclosure contains many specific implementation details, these should not be construed as limitations on the scope of any inventions or of what may be claimed, but rather as descriptions of features specific to particular embodiments. It should be understood that various modifications can be made without departing from the spirit and scope of the invention. The logic flows depicted in the figures do not require the particular order shown, or sequential order, to achieve desirable results. In addition, other or additional steps, elements, or connections can be included, or some of the steps, elements, or connections can be eliminated, or a combination thereof can be utilized in the described flows, illustrations, or descriptions. Further, the various aspects of the disclosed devices, apparatuses, systems, and/or methods can be combined in whole or in part with each other to produce additional implementations. Moreover, separation of various components in the embodiments described herein should not be understood as requiring such separation in all embodiments, and it should be understood that the described components can generally be integrated together in a single software product or packaged into multiple software products. Accordingly, other embodiments are within the scope of the following claims.

The invention claimed is:

1. A system comprising:

one or more processors; and

one or more non-transitory machine readable media storing machine readable code that, when executed by the one or more processors, causes the one or more processors to perform at least:

receiving or generating a first one or more digital pictures, wherein the first one or more digital pictures depict at least a portion of a first device's surrounding;

receiving or generating a first one or more instruction sets for operating the first device; and

learning the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device.

2. The system of claim **1**, wherein the receiving the first one or more digital pictures includes receiving the first one or more digital pictures from a picture capturing apparatus, and wherein the receiving the first one or more instruction sets for operating the first device includes receiving the first one or more instruction sets for operating the first device from: an application for operating the first device, a system for operating the first device, one or more microcontrollers, another one or more processors, or one or more actuators, and wherein the learning the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device includes storing the first one or more digital pictures correlated with the first one or more

instruction sets for operating the first device into or onto at least one of: at least one non-transitory machine readable medium of the one or more non-transitory machine readable media, another one or more non-transitory machine readable media, one or more volatile memories, one or more non-volatile memories, one or more storage devices, or one or more storage systems.

3. The system of claim **1**, wherein the learning the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device includes determining that the first one or more instruction sets for operating the first device temporally correspond to the first one or more digital pictures.

4. The system of claim **1**, wherein the machine readable code, when executed by the one or more processors, causes the one or more processors to further perform at least:

receiving or generating a new one or more digital pictures;

determining the first one or more instruction sets for operating the first device based on at least partial match between the new one or more digital pictures and the first one or more digital pictures; and

at least in response to the determining, causing the first device or a second device to perform one or more operations defined by the first one or more instruction sets for operating the first device.

5. The system of claim **4**, wherein the causing the first device or the second device to perform the one or more operations defined by the first one or more instruction sets for operating the first device includes executing the first one or more instruction sets for operating the first device.

6. The system of claim **4**, wherein the determining the first one or more instruction sets for operating the first device based on the at least partial match between the new one or more digital pictures and the first one or more digital pictures includes determining at least partial match between one or more portions of the new one or more digital pictures that represent one or more objects and one or more portions of the first one or more digital pictures that represent one or more objects.

7. The system of claim **4**, wherein the determining the first one or more instruction sets for operating the first device based on the at least partial match between the new one or more digital pictures and the first one or more digital pictures includes determining at least partial match between one or more objects detected in the new one or more digital pictures and one or more objects detected in the first one or more digital pictures.

8. The system of claim **4**, wherein the first one or more instruction sets for operating the first device include one or more information about one or more states of: the first device, or a portion of the first device.

9. The system of claim **4**, wherein the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device are included in a knowledgebase, and wherein the knowledgebase further includes a second one or more digital pictures correlated with a second one or more instruction sets for operating the first device, and wherein at least a portion of the first one or more digital pictures or at least a portion of the first one or more instruction sets for operating the first device is learned in a first learning process that includes operating the first device at least partially by a first user, and wherein at least a portion of the second one or more digital pictures or at least a portion of the second one or more instruction sets for operating the first device is learned in a second learning process that includes operating the first device at least partially by the first user.

10. The system of claim **4**, wherein the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device are included in a knowledgebase, and wherein the knowledgebase further includes a second one or more digital pictures correlated with a second one or more instruction sets for operating the first device, and wherein at least a portion of the first one or more digital pictures or at least a portion of the first one or more instruction sets for operating the first device is learned in a first learning process that includes operating the first device at least partially by a first user, and wherein at least a portion of the second one or more digital pictures or at least a portion of the second one or more instruction sets for operating the first device is learned in a second learning process that includes operating the first device at least partially by a second user.

11. The system of claim **4**, wherein the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device are included in a knowledgebase, and wherein the knowledgebase further includes a second one or more digital pictures correlated with a second one or more instruction sets for operating a third device, and wherein at least a portion of the first one or more digital pictures or at least a portion of the first one or more instruction sets for operating the first device is learned in a first learning process that includes operating the first device at least partially by a first user, and wherein at least a portion of the second one or more digital pictures or at least a portion of the second one or more instruction sets for operating the third device is learned in a second learning process that includes operating the third device at least partially by: the first user, or a second user.

12. The system of claim **4**, wherein the new one or more digital pictures depict at least a portion of the first device's surrounding, and wherein the first one or more instruction sets for operating the first device are applied to the first device, and wherein the first device is caused to perform the one or more operations defined by the first one or more instruction sets for operating the first device.

13. The system of claim **4**, wherein the new one or more digital pictures depict at least a portion of the second device's surrounding, and wherein the first one or more instruction sets for operating the first device are applied to the second device, and wherein the second device is caused to perform the one or more operations defined by the first one or more instruction sets for operating the first device.

14. The system of claim **4**, wherein the new one or more digital pictures depict at least a portion of the second device's surrounding, and wherein the first one or more instruction sets for operating the first device or a copy of the first one or more instruction sets for operating the first device are modified and applied to the second device, and wherein the second device is caused to perform one or more operations defined by: the modified the first one or more instruction sets for operating the first device, or the modified the copy of the first one or more instruction sets for operating the first device.

15. The system of claim **4**, wherein the machine readable code, when executed by the one or more processors, causes the one or more processors to further perform at least:

modifying:

the first one or more instruction sets for operating the first device, or

a copy of the first one or more instruction sets for operating the first device, and wherein the determining the first one or more instruction sets for operating the first device based on the at least partial match between the new one or more digital pictures and the first one or more digital pictures includes:

determining the modified the first one or more instruction sets for operating the first device based on the at least partial match between the new one or more digital pictures and the first one or more digital pictures, or

determining the modified the copy of the first one or more instruction sets for operating the first device based on the at least partial match between the new one or more digital pictures and the first one or more digital pictures, and wherein the causing the first device or the second device to perform the one or more operations defined by the first one or more instruction sets for operating the first device includes:

causing the first device or the second device to perform one or more operations defined by the modified the first one or more instruction sets for operating the first device, or

causing the first device or the second device to perform one or more operations defined by the modified the copy of the first one or more instruction sets for operating the first device.

16. The system of claim **4**, wherein the machine readable code, when executed by the one or more processors, causes the one or more processors to further perform at least:

modifying at least one of: the first one or more digital pictures, a copy of the first one or more digital pictures, the new one or more digital pictures, or a copy of the new one or more digital pictures, and wherein the at least partial match between the new one or more digital pictures and the first one or more digital pictures includes: (i) at least partial match between the modified the new one or more digital pictures and the first one or more digital pictures, (ii) at least partial match between the modified the copy of the new one or more digital pictures and the first one or more digital pictures, (iii) at least partial match between the new one or more digital pictures and the modified the first one or more digital pictures, (iv) at least partial match between the new one or more digital pictures and the modified the copy of the first one or more digital pictures, (v) at least partial match between the modified the new one or more digital pictures and the modified the first one or more digital pictures, (vi) at least partial match between the modified the copy of the new one or more digital pictures and the modified the copy of the first one or more digital pictures, (vii) at least partial match between the modified the new one or more digital pictures and the modified the copy of the first one or more digital pictures, or (viii) at least partial match between the modified the copy of the new one or more digital pictures and the modified the first one or more digital pictures.

17. The system of claim **4**, wherein the machine readable code, when executed by the one or more processors, causes the one or more processors to further perform at least:

modifying:

the first one or more digital pictures, or

a copy of the first one or more digital pictures, and wherein the learning the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device includes:

learning the modified the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device, or

learning the modified the copy of the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device, and wherein the determining the first one or more instruction sets for operating the first device based on the at least partial match between the new one or more digital pictures and the first one or more digital pictures includes:

determining the first one or more instruction sets for operating the first device based on at least partial match between the new one or more digital pictures and the modified the first one or more digital pictures, or

determining the first one or more instruction sets for operating the first device based on at least partial match between the new one or more digital pictures and the modified the copy of the first one or more digital pictures.

18. The system of claim **4**, wherein the machine readable code, when executed by the one or more processors, causes the one or more processors to further perform at least:

modifying:

the first one or more instruction sets for operating the first device, or

a copy of the first one or more instruction sets for operating the first device, and wherein the learning the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device includes:

learning the first one or more digital pictures correlated with the modified the first one or more instruction sets for operating the first device, or

learning the first one or more digital pictures correlated with the modified the copy of the first one or more instruction sets for operating the first device, and wherein the determining the first one or more instruction sets for operating the first device based on the at least partial match between the new one or more digital pictures and the first one or more digital pictures includes:

determining the modified the first one or more instruction sets for operating the first device based on the at least partial match between the new one or more digital pictures and the first one or more digital pictures, or

determining the modified the copy of the first one or more instruction sets for operating the first device based on the at least partial match between the new one or more digital pictures and the first one or more digital pictures, and wherein the causing the first device or the second device to perform the one or more operations defined by the first one or more instruction sets for operating the first device includes:

causing the first device or the second device to perform one or more operations defined by the modified the first one or more instruction sets for operating the first device, or

causing the first device or the second device to perform one or more operations defined by the modified the copy of the first one or more instruction sets for operating the first device.

19. The system of claim **4**, wherein the system further comprising:

an artificial intelligence system, and wherein the learning the first one or more digital pictures correlated with the first one or more instruction sets for operating the first

device includes learning, at least in part by the artificial intelligence system, the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device, and wherein the determining the first one or more instruction sets for operating the first device based on the at least partial match between the new one or more digital pictures and the first one or more digital pictures includes determining, at least in part by the artificial intelligence system, the first one or more instruction sets for operating the first device based on the at least partial match between the new one or more digital pictures and the first one or more digital pictures.

20. The system of claim **19**, wherein the artificial intelligence system includes: one or more inputs for receiving one or more digital pictures, and one or more outputs for providing one or more instruction sets, and wherein the learning, at least in part by the artificial intelligence system, the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device includes applying the first one or more digital pictures to the inputs of the artificial intelligence system, and wherein the determining, at least in part by the artificial intelligence system, the first one or more instruction sets for operating the first device based on the at least partial match between the new one or more digital pictures and the first one or more digital pictures includes:

applying the new one or more digital pictures to the inputs of the artificial intelligence system; and

receiving the first one or more instruction sets for operating the first device from the outputs of the artificial intelligence system.

21. The system of claim **20**, wherein the one or more inputs for receiving the one or more digital pictures include: one input for one digital picture, multiple inputs for multiple digital pictures, one input for one portion of one digital picture, multiple inputs for multiple portions of one digital picture, multiple inputs for multiple portions of multiple digital pictures, an input for one representation of one digital picture, multiple inputs for multiple representations of multiple digital pictures, one input for one representation of one portion of one digital picture, multiple inputs for multiple representations of multiple portions of one digital picture, or multiple inputs for multiple representations of multiple portions of multiple digital pictures.

22. The system of claim **19**, wherein the artificial intelligence system includes: a knowledgebase, or a neural network.

23. The system of claim **4**, wherein the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device include the first one or more digital pictures connected, using at least one or more connections, with the first one or more instruction sets for operating the first device.

24. The system of claim **4**, wherein the first one or more digital pictures are: one or more whole digital pictures, one or more representations of one or more whole digital pictures, one or more portions of at least one digital picture, one or more representations of one or more portions of at least one digital picture, one or more features, one or more representations of one or more features, one or more collections of pixels, or one or more collections of values, and wherein the new one or more digital pictures are: one or more whole digital pictures, one or more representations of one or more whole digital pictures, one or more portions of at least one digital picture, one or more representations of one or more portions of at least one digital picture, one or more features, one or more representations of one or more features, one or more collections of pixels, or one or more collections of values.

25. The system of claim **4**, wherein the system further comprising:

a server that receives from the first device at least one of: the first one or more digital pictures, or the first one or more instruction sets for operating the first device, and wherein the second device receives from the server at least one of: the first one or more digital picture % or the first one or more instruction sets for operating the first device, and wherein the second device is caused to perform the one or more operations defined by the first one or more instruction sets for operating the first device.

26. The system of claim **4**, wherein at least some elements of the system are included in: a single device, or multiple devices, and wherein the one or more processors include: one or more microcontrollers, one or more computing devices, or one or more electronic devices, and wherein the first device includes a robot, a vehicle, an appliance, an electronic device, or a mechanical machine, and wherein the second device includes a robot, a vehicle, an appliance, an electronic device, or a mechanical machine, and wherein the first one or more digital pictures include: one or more still digital pictures, or one or more motion digital pictures, and wherein the new one or more digital pictures include: one or more still digital pictures, or one or more motion digital pictures, and wherein an instruction set of the first one or more instruction sets for operating the first device includes at least one of: only one instruction, a plurality of instructions, one or more inputs, one or more commands, one or more keywords, one or more symbols, one or more operators, one or more variables, one or more parameters, one or more characters, one or more numbers, one or more values, one or more signals, one or more binary bits, one or more functions, one or more function references, one or more objects, one or more object references, one or more data structures, one or more data structure references, one or more states, one or more representations of one or more states, one or more representations of one or more user inputs, one or more codes, one or more data, or one or more information.

27. The system of claim **4**, wherein the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device are included in a knowledgebase, and wherein the knowledgebase further includes a second one or more digital pictures correlated with a second one or more instruction sets for operating a third device, and wherein the machine readable code, when executed by the one or more processors, causes the one or more processors to further perform at least:

receiving or generating an additional one or more digital pictures;

determining the second one or more instruction sets for operating the third device based on at least partial match between the additional one or more digital pictures and the second one or more digital pictures; and

at least in response to the determining the second one or more instruction sets for operating the third device, causing a fourth device to perform one or more operations defined by the second one or more instruction sets for operating the third device.

28. The system of claim **4**, wherein the at least partial match between the new one or more digital pictures and the first one or more digital pictures is at least in part based on

a product of one or more values that represent one or more portions of the new one or more digital pictures and one or more values that represent one or more portions of the first one or more digital pictures.

29. The system of claim **4**, wherein the generating the new one or more digital pictures includes:

detecting one or more objects in the first device's surrounding or the second device's surrounding;

generating one or more representations of the one or more objects; and

generating the new one or more digital pictures that include the one or more representations of the one or more objects.

30. The system of claim **29**, wherein the detecting the one or more objects in the first device's surrounding or the second device's surrounding includes detecting at least one or more locations of the one or more objects, and wherein the new one or more digital pictures include a top-down view of the one or more representations of the one or more objects.

31. The system of claim **4**, wherein the generating the first one or more instruction sets for operating the first device includes generating the first one or more instruction sets for operating the first device as one or more representations of a second one or more instruction sets for operating the first device.

32. The system of claim **4**, wherein the first one or more digital pictures that depict the at least the portion of the first device's surrounding include one or more digital pictures that depict a representation of the at least the portion of the first device's surrounding.

33. A method implemented using a computing system that includes one or more processors, the method comprising:

receiving or generating a first one or more digital pictures, wherein the first one or more digital pictures depict at least a portion of a first device's surrounding;

receiving or generating a first one or more instruction sets for operating the first device; and

learning the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device.

34. A system comprising:

means for receiving or generating a first one or more digital pictures, wherein the first one or more digital pictures depict at least a portion of a first device's surrounding;

means for receiving or generating a first one or more instruction sets for operating the first device; and

means for learning the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device.

* * * * *

Exhibit F

IN THE UNITED STATES DISTRICT COURT
FOR THE DISTRICT OF DELAWARE

AUTONOMOUS DEVICES LLC, Plaintiff, vs. TESLA, INC., Defendant.	Civil Action No. 22-1466-MN JURY TRIAL DEMANDED

DECLARATION OF ELI SABER, PH.D.

I, Eli Saber, Ph.D., hereby declare the following:

I. INTRODUCTION

1. Autonomous Devices retained me to offer technical opinions relating to U.S. Patent Nos. 10,452,974 ("the '974 Patent"); 11,238,344 ("the '344 Patent"); 11,663,474 ("the '474 Patent") 10,607,134 ("the '134 Patent"); 11,113,585 ("the '585 Patent"); 11,055,583 ("the '583 Patent"); and 10,102,449 ("the '449 Patent") (collectively, the "Asserted Patents").

2. I am being compensated for my work on this matter by Autonomous Devices. My compensation is not dependent on the contents of this Declaration, the substance of any further opinions I may offer, or the outcome of this matter.

3. I have been asked to give my opinion on what the Asserted Patents are directed to and whether they are novel over the prior art. As explained in detail below, it is my opinion that the claims of each of the Asserted Patents are directed to resolving specific issues in artificial intelligence (AI), autonomous driving, and/or simulation technology and that each recites a sufficient "inventive concept" such that is novel over the prior art.

4. I reserve whatever right I may have to supplement this Declaration if further information becomes available or if I am asked to consider additional information. Furthermore, I reserve whatever right I may have to consider and comment on any additional expert statement and testimony of the Defendants' experts in this matter.

II. QUALIFICATIONS

5. My curriculum vitae, including my qualifications and a list of publications I have authored, is attached to this Declaration as Appendix A.

6. I received a Ph.D. in Electrical Engineering from the University of Rochester in 1996. My concentration was on Signal/Image/Video Processing, Pattern Recognition, and Computer Vision. I received a Master's Degree in Electrical Engineering from the University of

Rochester in 1992, and I received a Bachelor's Degree in Electrical and Computer Engineering, Summa Cum Laude, from the State University of New York at Buffalo in 1998.

7. I am a professor in the Electrical and Microelectronic Engineering Department and the Chester F. Carlson Center for Imaging Science at the Rochester Institute of Technology (RIT) in Rochester, New York. I also serve as the Director of the Image, Video and Computer Vision Laboratory. I joined RIT's full-time faculty in 2004. I have 34 years of industry and academic experience, 30 of which are in the field of imaging.

8. Before becoming a full-time professor, I worked for Xerox Corporation for 16 years, from 1988 to 2004. During my years at Xerox, I was responsible for delivering color management, image processing innovations, architectures, and algorithms; xerographic sub-systems for a variety of color products; and control systems for toner production facilities. One of several roles I held at Xerox was Advanced Development Scientist and Manager. In that capacity, I established the Advanced Design Laboratory – an Imaging/Xerographics lab – and provided technical and managerial leadership for the Electrical, Imaging and Xerographics Department. In another role, as Product Development Scientist and Manager, I led the research and development of image quality metrics for various product platforms. I also led the Image Science, Analysis, and Evaluation area, with 12 to 15 direct reports and a budget of approximately $2 million.

9. From 1997 until 2004, I was an adjunct faculty member in the Electrical Engineering Department of RIT and in the Electrical and Computer Engineering Department of the University of Rochester. I was responsible for teaching undergraduate and graduate coursework in signal, image and video processing; pattern recognition; and communications.

Additionally, I performed research in multimedia applications, computer vision, pattern recognition, image understanding and color engineering.

10. Since joining RIT full-time in 2004, I have been responsible for teaching a variety of undergraduate and graduate courses in digital signal processing, digital image processing, digital video processing (a course that I have founded and pioneered at RIT), engineering analysis, random signal and noise, advanced engineering mathematics, matrix methods, pattern recognition (a course that I founded and pioneered at RIT), communications, modern control theory, and linear systems.

11. My current research focuses on developing image/video processing and computer vision algorithms for multimedia and military applications. To this end, I have extensive experience in developing digital image/video processing and computer vision algorithms and techniques for motion estimation, segmentation, registration, classification, identification, pattern recognition, facial recognition, biomedical image processing, compression, surveillance systems, multiple camera systems, machine learning, deep learning and artificial intelligence and the fusion of techniques for a variety of applications including object recognition and tracking, video analysis, video surveillance, change detection, biomedical, color engineering, and document processing using multimodal/multispectral type imagery.

12. As a principal investigator or co-principal investigator, I have acquired research funding in excess of $5 million since joining RIT and have managed multiple government grants from the Department of Defense as well as several corporate grants from Hewlett-Packard, Lenel, and Data Physics. I am currently managing, as principal investigator, one government grant from the Department of Defense for “Target Detection/Tracking and Activity Recognition from Multimodal data”. The work involves the development of machine learning/deep

learning/artificial intelligence type algorithms for fusion of multiple modalities using convolutional neural networks (CNN) to detect and tract small/sparse targets in aerial imagery.

13. In 2012, I was awarded the Prestigious Trustees Scholarship, the highest award at RIT with regard to research recognition

14. I am a senior member of the Institute of Electrical and Electronic Engineers and a member of the IEEE Signal Processing Society, the Electrical Engineering Honor Society, and Eta Kappa Nu.

15. I am the author or co-author of 38 peer-reviewed journal publications.

16. I have also authored or co-authored 100 conference and workshop publications and a book entitled Advanced Linear Algebra for Engineers with MATLAB, published by CRC Press in February 2009, and am a named inventor on multiple U.S. and foreign patents.

III. PERSON OF ORDINARY SKILL IN THE ART

17. In my opinion, a person of ordinary skill in the art ("POSITA") in the field of the Asserted Patents would have at least: (1) a bachelor's degree in electrical and/or computer engineering, or computer science (or equivalent course work) with two to three years of work experience in computer vision, image/video processing and the design and development of neural networks; or (2) a master's degree in electrical and/or computer engineering, or computer science (or equivalent course work) with a focus in computer vision, image/video processing and the design and development of neural networks. The POSITA would have gained this knowledge through a combination of education and work experience, for example, a master's degree in a relevant engineering field, such as electrical or computer engineering or computer science, and at least three years of work experience in that field. More education can substitute for less work experience, and more work experience can substitute for less education.

IV. MATERIALS CONSIDERED

18. I have reviewed the Asserted Patents, the prosecution history of each Asserted Patent, and other materials referenced in Appendix B to this Declaration. Counsel has informed me that I should consider these materials through the lens of a POSITA at the time of the priority dates of the patents, and I have done so. For the ’974 patent, the ’344 patent, and the ’474 patent, I have assumed the priority date is November 2, 2006. For the ’134 patent, I have assumed the priority date is December 19, 2016. For the ’585 patent, I have assumed the priority date is August 23, 2016. For the ’583 patent, I have assumed the priority date is November 26, 2017. For the ’449 patent patent, I have assumed the priority date is November 21, 2017.

19. My analyses are based on my education and work experience and my investigation and study of materials listed in Appendix B.

V. APPLICABLE LEGAL PRINCIPLES

20. Although I am not an attorney, my understanding of the relevant law is as follows. I have been informed that a federal statute, 35 U.S.C. § 101, describes the types of inventions that are patentable. I have been further informed that the Supreme Court has held that this provision contains an important explicit exception: Laws of nature, natural phenomena, and abstract ideas are not patentable.

21. In its 2014 decision in *Alice Corp. v. CLS Bank Int’l*, I understand that the Supreme Court outlined a two-step test to determine whether a patent is ineligible under § 101 because it is directed to an abstract idea. 573 U.S. 208 (2014).

22. I have been informed that in the first step, the court determines whether the claims at issue are directed to a patent-ineligible concept, such as an abstract idea. I understand that federal courts have identified several factors to help determine whether an invention is directed to an abstract idea in the context of computer-related inventions.

23. I have been informed, for example, that courts examine whether a claim is directed to specific asserted improvements in computer capabilities or whether a claim is instead directed to a process that qualifies as an abstract idea for which computers are invoked as a tool. I understand that in the former case, the patent would not be directed to an abstract idea but in the latter case, it would.

24. I understand that courts have also phrased the step one inquiry to ask whether the claims merely implement an old practice in a new environment.

25. I have been informed that if the court determines the claims at issue are directed to an abstract idea, the court proceeds to a second step, asking what is in the claims before it. I am informed that in step two the court examines the elements of the claim to determine whether it contains an inventive concept sufficient to transform the claim abstract idea into a patent eligible application. I am informed that a court determines whether the claim elements individually, or as an ordered combination, contain an inventive concept which is more than merely implementing an abstract idea using well-understood, routine and conventional activities previously known in the industry. I understand that this step is often described as a search for an "inventive concept"—*i.e.*, an element or combination of elements sufficient to ensure that the patent in practice amounts to significantly more than a patent upon the ineligible concept.

26. I understand that whether a claim recites patent-eligible subject matter is a question of law that may contain underlying facts.

27. I understand that federal courts have said that whether a claim element or combination of elements is well-understood, routine, and conventional to a skilled artisan in the relevant field is a question of fact.

VI. OPINIONS

28. Since well before the priority dates of the Asserted Patents, the pursuit of autonomous driving has been extremely important. In fact, achieving a driverless vehicle has been the holy grail of the automotive industry.[1] Companies, such as Tesla, have spent an enormous sum of money in the pursuit of autonomous vehicles. Based on my experience in computer vision, machine learning, and computer assisted machine operations, achieving autonomous driving has been a significant problem due in large part to the complexity of the "driving" environment and the near-infinite number of circumstances encountered by such a vehicle.

29. Part of the difficulty was that prior art driver assistance aids were coded into the system, e.g., keep a certain distance from the lead vehicle,[2] maintain the vehicle between the lines on the road,[3] etc. Coding responses to circumstances was not sufficiently precise to enable autonomous device operation across the vast array of circumstances a vehicle may come across on a daily basis. And known "techniques lack[ed] a way to ***learn*** [the] operation of a device" and share that learned operation with another device to enable its autonomous operation. '974 patent at 1:26-35.

30. For example, the Notice of Allowance for the '974 patent stated that "the prior art does not disclose [a] device that operates as the user directs it, hence, does not disclose ***learning the device operation from the user directing the device***." '974 patent Notice of Allowance at 14; *see also* '344 patent Notice of Allowance at 12 (reciting similar reasons). The Notice of Allowance highlighted a robot patent (U.S. 2016/0167226) and its failure to disclose user

---

[1] https://www.bloomberg.com/news/articles/2021-11-18/apple-accelerates-work-on-car-aims-for-fully-autonomous-vehicle

[2] http://sunnyday.mit.edu/safety-club/workshop5/Adaptive_Cruise_Control_Sys_Overview.pdf

[3] https://www.nissan-global.com/EN/INNOVATION/TECHNOLOGY/ARCHIVE/LDW/

assisted learning together with the remaining claimed features. That robot constructs a map of its environment using V-SLAM (Visual Simultaneous Location and Mapping). U.S. 2016/0167226 at ¶99. Using V-SLAM, the robot can "track its location and establish ground truth information with respect to captured images and/or image portions corresponding to views of specific map locations." *Id.* But there is no learning based on user operation and no sharing of learned responses to circumstances with other devices. *See, e.g.*, '974 patent at 1:26-35 ("[c]ommonly employed device or system operating techniques lack[ed] a way to ***learn [the] operation of a device***" to "enable autonomous operation" of that device and/or another device); '449 Patent at 1:20-23 (known "devices or systems [were] limited to relying on the user to direct them."); '583 Patent at 1:31-39 ("These systems and/or devices depend on user's input to various degrees for their operation. A machine learning solution [was] needed for computing enabled systems and/or devices to be less dependent on or fully independent from user input.").

31. The Notice of Allowance further explained that the "[p]rior art discloses a system that explores its own environment on its own, which is very different from relying on the user to direct it." '974 Notice of Allowance at 14; *see also* '344 patent Notice of Allowance at 12 (reciting similar reasons). And the "[p]rior art does not disclose the first and the second correlations that each include a circumstance representation correlated with instruction sets." '974 patent Notice of Allowance at 14; *see also* '344 patent Notice of Allowance at 12 (reciting similar reasons). Thus, the "[p]rior art does not disclose the knowledgebase," which includes a "circumstance representation (a data structure itself)," an "instruction set (a data structure itself when learned in the learning process)" or a "correlation (a data structure itself) that includes a circumstance representation correlated with instruction sets." '974 patent Notice of Allowance at 14-15; *see also* '344 patent Notice of Allowance at 12 (reciting similar reasons).

32. Similarly, in allowing the ’583 patent, the USPTO explained that the prior art fails to disclose learning the correlation between an image and an instruction set—rather the prior art “only adds [an] unidentified object to the database by a manual or formal approval process” and there is no suggestion of “learning a correlation between the image and any instruction set(s) for operating a device.” ’583 patent Notice of Allowance at 12-13.

33. Notably, early prior art systems attempted to achieve vehicle self-driving by training the system using synthetic road images, which resulted in poor performance in real driving situations. *See* Pomerleau, D.A. (1991) Efficient Training of Artificial Neural Networks for Autonomous Navigation in Neural Computation, pp. 88-97 at 4. To overcome these problems, the next generation prior art systems tried to imitate a person’s driving under actual driving conditions. *Id.* at 5. However, those systems suffered from flaws due to overlearning from repetitive inputs. *Id.* For example, if the driver goes down a straight stretch of road during part of a training run, the network was presented with a long sequence of similar images. *Id.* This sustained lack of diversity in the training caused the network to “forget” what it had learned about driving on curved roads and instead learn to, generally speaking, steer straight ahead. *Id.*

34. More recent prior art suggests that carmakers have resorted to more predictable and less robust solutions. For example, known lane-keeping assist (“LKA”) systems detect lane markers (white/yellow) on the road and assist the driver in keeping the vehicle between lane markers.[4] Another example of a known driver assistance feature is adaptive cruise control (“ACC”). Adaptive cruise control measures the distance from the vehicle ahead and controls the acceleration and deceleration of the vehicle at hand to automatically maintain a suitable

[4] https://www.nissan-global.com/EN/INNOVATION/TECHNOLOGY/ARCHIVE/LDW/

following distance.[5] These systems, however, simply executed pre-coded responses to a limited number of circumstances.

35. As a result, there has long been a demand for solutions that can account for the broad array of scenarios encountered by an autonomously operating device. The innovations in the Asserted Patents, as discussed below, permit devices to operate autonomously at least based on learned experiences from other devices. These innovations are novel and inventive. Said another way, the prior art systems lacked the ability to *learn from a driver's response to a circumstance and share learned responses across a fleet of vehicles so that those vehicles can respond autonomously to a similar situation*.

36. The prior art's limited capabilities are not surprising. Generally speaking, the art had moved away from early attempts at neural network-based algorithms and towards coded responses to limited situations in order to enable solutions such as LKA and/or ACC. The reasons for this transition are numerous. To this effect, early attempts at using neural network-based solutions for self-driving were known to have problems with overlearning from their most recent experience(s). Moreover, complex environments, such as busy city streets or highways, offer many dynamic changes and challenges that need to be resolved in real or near real time in order to render proper actions. Hence, a neural network-based solution would have required a high-degree of complexity with many potential nodes/layers to accurately and effectively manage the challenges and dynamics of the environment; and ample computational power to compute solutions/render actions in real or near real time. However, until recently, the computation engines required to host the neural network-based solutions were not up to par to handle the complexity and dynamics of the underlying environment. With the

[5] http://sunnyday.mit.edu/safety-club/workshop5/Adaptive_Cruise_Control_Sys_Overview.pdf

introduction/arrival of more powerful processors and multitasking capabilities, many new doors were opened, such as those that led to the innovations in the Asserted Patents.

37. Given the difficulties with early neural network applications and the limitations with pre-coded solutions, it was not known or obvious to use a fleet of vehicles to train one another to react autonomously to a diverse set of situations.

A. THE OBJECT REPRESENTATION PATENTS (US Patent Nos. 10,452,974, 11,238,344, and 11,663,474)

38. The Object Representation Patents are generally directed to a system, method and computer readable medium. By way of example, the system stores a knowledgebase that correlates device operating instruction sets with circumstances encountered by a device, obtains a new circumstance representation using device sensors, matches the new circumstance with a circumstance in an a priori maintained knowledgebase, and causes the device that detected the new circumstance to autonomously execute an instruction set correlated to the matched circumstance. The autonomously performed operation is based on a correlation previously acquired in a learning process that includes operating, at least partially by a user, the device that captured the instruction and the stored circumstance.

39. For example, if a second autonomous vehicle detects a circumstance representation (e.g., one or more object representations such as debris or people in the road) and the knowledgebase has a similar circumstance representation that matches with the detected circumstance, then the second autonomous vehicle performs an instruction set (e.g., braking) at least partially learned by operating a first autonomous vehicle that captured the similar circumstance representation.

40. Referring to Figure 38, for example, vehicle 98 includes sensors 92, such as a camera 92a that detect objects 615aa-ad (people, boulders, materials, vehicles, etc.) in the

vehicle's 98a's surrounding in the form of a circumstance representation.[6] '344 Patent[7] at 162:48-163:16; Fig. 38. If the vehicle's computer(s) detect a match (or partial match) with a previously learned circumstance representation 525 (e.g., one or more object representations), the vehicle 98 performs autonomous operation using previously learned instructions 526 correlated with the previously learned circumstance representation. *Id.* at 163:36-164:5.

41. In my opinion, the corresponding claimed features were not well-understood, routine or conventional as of the priority date of the Object Representation Patents. For instance, the memory of the device performing the autonomous operation stores a knowledgebase that correlates instructions for performing the operation with the stored circumstance representation.[8] The device performing the autonomous operation also obtains a circumstance representation while in operation, e.g., it captures circumstances surrounding a vehicle using its camera, for example.[9] When there is at least a partial match between the stored circumstance representation

---

[6] Figure citations include the specification discussion of each figure.

[7] The Object Representation Patents share a specification. For simplicity, duplicative citations are not always provided.

[8] *See e.g.*, '974 patent at cl. 1 ("a memory that stores at least a knowledgebase that includes: a first correlation including a first circumstance representation correlated with a first one or more instruction sets …"), cl. 18 ("accessing a memory that stores at least a knowledgebase that includes: a first correlation including a first circumstance representation correlated with a first one or more instruction sets for operating a first device"); '344 patent at cl. 1 ("a memory that stores at least a knowledgebase that includes a first circumstance representation correlated with a first one or more instruction sets for operating a first device…"); '474 patent at cl. 45 ("a knowledgebase that includes one or more inputs for inputting at least a portion of a circumstance representation, wherein the one or more inputs are correlated with one or more instruction sets for operating a second device").

[9] *See e.g.*, '974 patent at cl. 1 ("the third circumstance representation represents a third circumstance detected … at least in part by one or more sensors of a second device"), cl. 18 ("the third circumstance representation represents a third circumstance detected … at least in part by one or more sensors of a second device"); '344 patent at cl. 1 ("the second circumstance representation represents a second circumstance detected at least in part by: … one or more sensors of a second device"); '474 patent at cl. 1 ("generating a circumstance representation, wherein the generated circumstance representation represents a circumstance detected at least in part by the one or more sensors").

and the current circumstance representation, the corresponding instruction is autonomously performed by the device that matched the current and stored circumstance representations.[10] The claim features linked to the above-described example are inventive and novel and, as of the priority date of the Object representation patents, were not well-understood, routine or conventional.

42. In some embodiments, the circumstance representations include one or more object representations, e.g., the circumstance representations included representations of object(s) such as debris, people, vehicles, etc.[11] The claim features linked to the above-described

[10] *See e.g.*, '974 patent at cl. 1 ("anticipating the first one or more instruction sets for operating the first device learned in the learning process based on at least partial match between the third circumstance representation and the first circumstance representation; and at least in response to the anticipating, executing the first one or more instruction sets for operating the first device learned in the learning process, wherein … the second device autonomously performs one or more operations defined by the first one or more instruction sets for operating the first device learned in the learning process"), cl. 18 ("anticipating the first one or more instruction sets for operating the first device learned in the learning process based on at least partial match between the third circumstance representation and the first circumstance representation; and at least in response to the anticipating, causing … the second device to perform one or more operations defined by the first one or more instruction sets for operating the first device learned in the learning process at least by causing the one or more processor circuits or another one or more processor circuits to execute the first one or more instruction sets for operating the first device learned in the learning process."); '344 patent at cl. 1 ("anticipating the first one or more instruction sets for operating the first device based on at least partial match between the second circumstance representation and the first circumstance representation; and at least in response to the anticipating, executing the first one or more instruction sets for operating the first device, wherein … the second device autonomously performs one or more operations defined by the first one or more instruction sets for operating the first device."); '474 patent at cl. 45 ("inputting at least a portion of the generated circumstance representation into the one or more inputs, and using a correlation between the one or more inputs and the one or more instruction sets for operating the second device; and at least in response to the determining, executing the one or more instruction sets for operating the second device, wherein the first device autonomously performs one or more operations defined by the one or more instruction sets for operating the second device.")

[11] *See, e.g.*, '344 patent at cl. 3 ("wherein the first circumstance representation includes a first one or more object representations, and wherein the second circumstance representation includes a second one or more object representations"); *see also* '974 patent at cl. 14 (" wherein the first circumstance representation includes: one or more object representations, or one or more

example are inventive and novel and, as of the priority date of the Object representation patents, were not well-understood, routine or conventional.

43. With the innovations in the Object Representation Patents, autonomous device operation using a knowledgebase (e.g., a neural network that includes, for example, a combination of circumstance/object recognition and correlated previously learned instructions) that is trained by a fleet of vehicles became possible.[12] *See, e.g.*,'344 Patent at 164:6-28; *See also id.* at 15:36-45; 38:54-40:34; 82:6-14; 95:15-96:25; 103:35-105:22; 108:40-44. The corresponding claimed features are inventive and novel and were not well-understood, routine or conventional as of the priority date of the Object Representation Patents.

44. By utilizing the autonomous device operating techniques disclosed in the Object Representation Patents, a vehicle can perform autonomous operations based on previously learned circumstances and instructions. For at least the above-mentioned reasons, the claimed inventions improve the capabilities of autonomous devices that employ the systems and methods disclosed in the Object Representation Patents. Further, the inventions claimed in the Object Representation Patents cannot be performed as mental steps by a human, nor do they represent the application of a generic computer to any well-known method of organizing human behavior.

---

collections of object representations, and wherein the second circumstance representation includes: one or more object representations, or one or more collections of object representations, and wherein the third circumstance representation includes: one or more object representations, or one or more collections of object representations."); '474 patent at cl. 46 (" at least the portion of the circumstance representation include one or more inputs for inputting at least a portion of one or more object representations, and wherein the generated circumstance representation includes one or more object representations…")

[12] *See e.g.*, '974 patent at cl. 1 (element starting with "at least in response to the anticipating, executing the first one or more instruction sets…"); '344 patent at cl. 1 (element starting with "at least in response to the anticipating, executing the first one or more instruction sets …"); '474 patent at cl. 45 ("executing the one or more instruction sets for operating the second device, wherein the first device autonomously performs one or more operations defined by the one or more instruction sets for operating the second device")

The claims of the Object Representation Patents are also directed to non-abstract ideas in that they provide technical solutions to at least the technical problems described above with respect to coded responses, lack of training set diversity, and user operation corresponding to the correlated circumstances and instructions in the knowledgebase.

45. Claim 1 of the ’974 and ’344 Object Representation Patents and claim 45 of the ’474 patent, as a whole, is inventive and novel, as are at least the herein identified claim limitations. *See, e.g.*,’974 Patent at Claim 1 (“accessing a memory that stores at least a knowledgebase that includes: a first correlation including a first circumstance representation correlated with a first one or more instruction sets for operating a first device”, “a learning process that includes operating the first device at least partially by a user” and “the second device autonomously performs one or more operations defined by the first one or more instruction sets for operating the first device learned in the learning process”), claim 18 (“a learning process that includes operating the first device at least partially by a user” and “causing … the second device to perform one or more operations defined by the first one or more instruction sets for operating the first device learned in the learning process at least by causing the one or more processor circuits or another one or more processor circuits to execute the first one or more instruction sets for operating the first device learned in the learning process.”);’344 Patent at Claim 1 (“accessing a memory that stores at least a knowledgebase that includes a first circumstance representation correlated with a first one or more instruction sets for operating a first device”, “a learning process that includes operating the first device at least partially by a user” and “the second device autonomously performs one or more operations defined by the first one or more instruction sets for operating the first device”); ’474 Patent at Claim 45 (“a knowledgebase that includes one or more inputs for inputting at least a portion of a circumstance representation,

wherein the one or more inputs are correlated with one or more instruction sets for operating a second device", "determining the one or more instruction sets for operating the second device at least by: inputting at least a portion of the generated circumstance representation into the one or more inputs, and using a correlation between the one or more inputs and the one or more instruction sets for operating the second device; and … wherein the first device autonomously performs one or more operations defined by the one or more instruction sets for operating the second device.")

46. . As of the priority date of the '974, '344, and '474 patents, the above-mentioned claim elements were not well-understood, routine, or conventional, and the novel user-trained knowledgebase that is capable of being deployed and utilized by other vehicles in the fleet is a vast improvement over the prior art.

47. Moreover, claim 3 of the '344 patent, claim 14 of the '974 patent and claim 46 of the '474 patent, as a whole, are also inventive and novel, as are at least the herein identified claim limitations. *See, e.g.*, '344 patent at cl. 3 ("wherein the first circumstance representation includes a first one or more object representations, and wherein the second circumstance representation includes a second one or more object representations"); *see also* '974 patent at cl. 14 (" wherein the first circumstance representation includes: one or more object representations, or one or more collections of object representations, and wherein the second circumstance representation includes: one or more object representations, or one or more collections of object representations, and wherein the third circumstance representation includes: one or more object representations, or one or more collections of object representations."); '474 patent at cl. 46 ("wherein the one or more inputs for inputting the at least the portion of the circumstance representation include one or more inputs for inputting at least a portion of one or more object

representations, and wherein the generated circumstance representation includes one or more object representations, and wherein the first device is a first vehicle, and wherein the second device is a second vehicle") As of the priority date of the '974, '344 and '474 patents, the above-mentioned claim elements were not well-understood, routine, or conventional, and the novel user-trained knowledgebase that is capable of being deployed and utilized by other vehicles in the fleet is a vast improvement over the prior art.

48. In my opinion, the above-mentioned claimed features unlocked the next level of autonomous driving because they allowed the entire fleet to train the system providing a higher degree to diversity in the training process. As a result, much more precise and appropriate reactions to driving circumstances became possible. This is because, using the claimed features, the massive number of scenarios encountered by the fleet could be distilled into a set of circumstances and corresponding instructions that can be efficiently performed autonomously by vehicles within the fleet. The corresponding claimed concepts were not known or done in the prior art.

49. The claims of the Object Representation Patents recite one or more inventive concepts rooted in computerized technology that overcome technical problems in that field. A person of ordinary skill in the art reading the Object Representation Patents and their claims would understand that the Object Representation Patents' disclosures and claims are drawn to solving specific technical problems arising in artificially intelligent and autonomous devices and systems. Accordingly, each claim of the Object Representation Patents recites a combination of elements sufficient to ensure that the claim in practice amounts to significantly more than a patent claiming an abstract concept. Further, the claimed improvements over the prior art are concrete and improve the capabilities of existing autonomous and artificial intelligence systems.

50. A person of ordinary skill in the art reviewing the specification of the Object Representation Patents would understand that the inventor had possession of the claimed subject matter and would know how to practice the claimed invention without undue experimentation.

B. THE DIGITAL PICTURE PATENTS (U.S. PATENT NOS. 10,102,449 & 11,055,583)

51. The ’583 Patent is generally directed to systems, methods, and non-transitory machine-readable mediums for correlating an instruction set for operating a first device with digital pictures, obtaining a new digital picture, and in response to matching the new digital picture with a correlated digital picture, causing the device that obtained the new digital picture to perform operations defined by the instruction set correlated to the matched digital picture.[13] The instruction set for operating the first device may be executed by a second device in response to the second device matching the new picture with the picture correlated to the instruction sets.[14] As of the priority date of the ’583 patent, the combination of the above-mentioned concepts corresponding to the claimed “learning,” obtaining “a new one or more digital pictures,” and “causing” a “second device to perform” an operation defined by the first instruction set(s) was not well-understood, routine or conventional.

52. Similarly, the ’449 Patent is generally directed to a system, method and non-transitory computer readable medium that correlates an instruction set for operating a first device with a digital picture, provides an artificial intelligence (AI) unit that: (i) receives a new digital picture, (ii) matches the new digital picture with the digital picture correlated to the instruction

[13] *See, e.g.*, ’583 patent at cl. 1 (“learning the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device”); cl. 4 (“receiving or generating a new one or more digital pictures”, “determining the first one or more instruction sets for operating the first device based on at least partial match between the new one or more digital pictures and the first one or more digital pictures”, “causing … a second device to perform one or more operations defined by the first one or more instruction sets for operating the first device.”)

[14] *Id.*

set, and (iii) causes a device that received the new digital picture to execute the instruction set correlated to the matched digital picture.[15] As of the priority date of the ’449 patent, the combination of the above-mentioned features corresponding to claim elements (a) to (e) was not well-understood, routine or conventional.

53. By way of example, consider a situation where a second autonomous vehicle receives a new digital picture of people in the road and where the knowledgebase has a similar digital picture that matches with the new digital picture. In this situation, using the above-discussed claimed inventions, the second autonomous vehicle performs an instruction set (e.g., braking) at least partially learned by operating a first autonomous vehicle. As of the priority date of the Digital Picture Patents, the execution of an instruction set on one vehicle where that instruction set was learned on another vehicle was inventive and novel, and this feature was not well-understood, routine, or conventional.

---

[15] *See, e.g.*, ’449 patent at cl. 1 (“a memory that stores at least a first one or more digital pictures correlated with a first one or more instructions sets for operating a first physical device, … a learning process that includes operating the first physical device at least partially by a user”, “an artificial intelligence unit that: receives a new one or more digital pictures from the optical camera; anticipates the first one or more instruction sets for operating the first physical device based on at least partial match between the new one or more digital pictures and the first one or more digital pictures, … causes the one or more processor circuits to execute the first one or more instruction sets for operating the first physical device, wherein the causes is performed in response to the anticipates of the artificial intelligence unit, and wherein … a second physical device autonomously performs one or more operations defined by the first one or more instruction sets for operating the first physical device.”), cl. 17 (“(a) accessing a memory that stores at least a first one or more digital pictures correlated with a first one or more instructions sets for operating a first physical device, … wherein at least a portion of the first one or more digital pictures or at least a portion of the first one or more instruction sets for operating the first physical device are learned in a learning process that includes operating the first physical device at least partially by a user”, “(b) receiving a new one or more digital pictures from an optical camera”, “(c) anticipating the first one or more instruction sets for operating the first physical device based on at least partial match between the new one or more digital pictures and the first one or more digital pictures”, “(d) executing the first one or more instruction sets for operating the first physical device,” and “(e) autonomously performing, … a second physical device, one or more operations defined by the first one or more instructions [sets] for operating the first physical device.”)

54. Referring to Figure 38 of the ’449 patent, for example, the vehicle 98a includes sensors 90, such as a camera 90a that detect the vehicle’s surroundings in the form of a digital picture. ’449 Patent at 152:21-154-27, Figs. 37, 38; *see also* ’583 Patent at 152:33-154:39, Figs. 37-38. If the vehicle’s computer(s) detect a match with a previously learned digital picture 525, the vehicle 98 performs autonomous operation using previously learned instructions 526 correlated with the previously learned digital picture. *See, e.g.*, ’449 Patent at 154:5-27; *see also* ’583 Patent at 154:14-24. The algorithms correlate instructions for performing an operation with the stored digital picture and when there is a match or partial match with the digital picture the operation is performed. ’449 Patent at 154:5-27; *see also* ’583 Patent at 154:14-24. The corresponding claimed features, as a combination, are inventive and novel and were not well-understood, routine or conventional as of the priority date of the Digital Picture Patents.

55. The memory of the device performing the autonomous operation stores a correlation between instructions for performing an operation and a digital picture.[16] When there is at least a partial match between the stored digital picture and a new digital picture, the operation is autonomously performed by the device that matched the new and stored digital pictures.[17] The corresponding claimed features are novel and inventive and were not well-understood, routine or conventional as of the priority date of the Digital Picture Patents.

56. With the innovations in the Digital Picture Patents, autonomous device operation using a memory with instruction and picture correlations (e.g., a neural network that includes, for example, a combination of digital pictures and correlated previously learned instructions) that is

[16] ’449 Patent at cl. 1 (element beginning with “a memory that stores...”), cl. 17 (element (a)); ’583 patent at claim 1 (element starting with “learning”); claim 4 (element starting with “determining”).

[17] ’449 patent at cl. 1 (elements beginning with “anticipates…” and “causes …”), cl. 17 (elements (c) to (e)); ’583 patent at claim 4 (element starting with “at least in response to”).

trained by a fleet of vehicles became possible. *See, e.g.*,'449 Patent at 14:38-45 ("In certain embodiments, the autonomous device operating includes a partially or a fully autonomous device operating. The partially autonomous device operating may include executing the one or more instruction sets for operating the device correlated with the first digital picture responsive to a user confirmation. The fully autonomous device operating may include executing the one or more instruction sets for operating the device correlated with the first digital picture without a user confirmation."); *see also id.* at Figs. 31-35; '583 Patent at 14:54-62, Figs. 31-35. The corresponding claimed features were not well-understood, routine or conventional as of the priority date of the Digital Picture Patents. As of the priority dates of the Digital Picture Patents, the above-mentioned claim elements were not well-understood, routine, or conventional, and the novel user-trained knowledgebase that is capable of being deployed and utilized by other vehicles in the fleet is a vast improvement over the prior art.

57. Thus, by utilizing the autonomous device operating techniques disclosed in the Digital Picture Patents, a vehicle can perform autonomous operations based on previously learned pictures and instructions. For at least the above-mentioned reasons, the claimed inventions improve the capabilities of autonomous devices that employ the systems and methods disclosed in the Digital Picture Patents. Further, the methods claimed in the Digital Picture Patents cannot be performed as mental steps by a human, nor do they represent the application of a generic computer to any well-known method of organizing human behavior. The claims of the Digital Picture Patents are also directed to non-abstract ideas in that they provide technical solutions to at least the technical problems described above with respect to coded responses, lack of training set diversity, and user operation corresponding to the correlated pictures and instructions in the knowledgebase.

58. The claimed features unlocked the next level of autonomous driving by allowing the entire fleet to train the overall system. In this way, much more precise and appropriate reactions to driving circumstances became possible because the massive number of scenarios encountered by the fleet could be distilled into a set of digital pictures and corresponding instructions that could be efficiently performed autonomously in an effective manner.

59. The claims of the Digital Picture Patents recite one or more inventive concepts rooted in computerized technology that overcome technical problems in that field. A person of ordinary skill in the art reading the Digital Picture Patents and their claims would understand that the Digital Picture Patents' disclosures and claims are drawn to solving specific technical problems arising in artificially intelligent and autonomous devices and systems. Accordingly, each claim of the Digital Picture Patents recites a combination of elements sufficient to ensure that the claim in practice amounts to significantly more than a patent claiming an abstract concept. Further, the claimed improvements over the prior art are concrete and improve the capabilities of existing autonomous and AI systems.

60. A person of ordinary skill in the art reviewing the specification of the Digital Picture Patents would understand that the inventor had possession of the claimed subject matter and would know how to practice the claimed invention without undue experimentation.

C. THE SIMULATION PATENTS (U.S. PATENT NOS. 10,607,134 & 11,113,585)

61. Claim 1 of both the Simulation Patents recites a system for causing an avatar or an object of an application program to autonomously perform operations defined by instruction sets previously learned by that device or another device by anticipating the instruction set(s) to be executed (e.g., apply brakes) in response to finding at least a partial match between a previously learned digital picture ('585 patent) or object representation ('134 patent) and a newly

received picture or object representation (e.g., similar debris on the road). *See* '134 patent at claim 1; '585 patent at claim 1.

62. Based on my extensive experience in computer vision, image/video processing, machine learning, and computer assisted machine operations, achieving methods for full-self driving has been difficult due in large part to the near-infinite number of circumstances encountered by a vehicle and the high-degree of variability has been a significant problem. For example, the prior art avatar and object operation was "semi-autonomous." As explained by the Patent Examiner, the prior art of record does not teach the details of object representations, avatars, determining instruction sets, and causing an avatar to perform operations as recited in the claims. *See generally* '134 Patent Notice of Allowance at 16. The examiner then went on to characterize certain prior art as teaching semi-autonomous avatar operation and learning, but not teaching how it was achieved. *Id.* Thus, as the specification states, "[c]ommonly employed application and/or object thereof operating techniques lack a way for a system to learn [the] operation of an application and/or object thereof and enable autonomous operation of an application and/or object thereof." '585 patent at 1:45-55; '134 patent at 1:30-40.

63. As of the priority date of the Simulation Patents, the claimed features in the Simulation Patents, as a whole, were inventive and novel and were not well-understood, routine, or conventional. In general, the Simulation Patents are directed to a system, method and computer readable medium that allows an avatar or object (i.e., a virtual car) in a program to autonomously perform operations in response to an acquired image or object representation (e.g., a current circumstance surrounding the avatar or device in the application/simulation as captured by the avatar's sensors, e.g., circumstance representations having object representations within

the circumstance such as debris or people in the road) at least partially matching an image or object representation stored in a memory.

64. The ’134 patent is specifically directed to innovations including correlating object representations with instruction sets for operating a first avatar of an application,[18] obtaining a second one or more object representations,[19] looking for a match between the second object representations and the first object representations,[20] and in response to finding a match causing a second avatar of the application to perform operations defined by the instruction set corresponding to the matched object representation.[21] For example, if a second avatar detects an object representation (e.g., objects such as virtual debris or virtual people in the road) and the system has a similar object representation that matches with the detected object, then the second avatar performs an instruction set (e.g., braking).[22] The corresponding claimed features, as a combination, are inventive and novel and were not well-understood, routine or conventional as of the priority date of the ’134 patent.

65. The ’585 patent is specifically directed to innovations including correlating digital pictures with instruction sets for operating a first object of an application,[23] obtaining a second

[18] ’134 patent at cl. 1 (“a first correlation including a first one or more object representations correlated with a first one or more instruction sets for operating a first avatar of an application”).
[19] ’134 patent at cl. 1 (“generating or receiving a second one or more object representations”).
[20] ’134 patent at cl. 1 (“determining the first one or more instruction sets for operating the first avatar of the application based on at least partial match between the second one or more object representations and the first one or more object representations”).
[21] ’134 patent at cl. 1 (“at least in response to the determining, causing … a second avatar of the application to perform one or more operations defined by the first one or more instruction sets for operating the first avatar of the application at least by executing the first one or more instruction sets for operating the first avatar of the application.”)
[22] *Id.*
[23] ’585 patent at cl. 1 (“one or more memories that store at least a first one or more digital pictures correlated with a first one or more instruction sets for operating a first object of a first application program”)

one or more digital pictures,[24] searching for a match between the second digital pictures and the first digital pictures,[25] and in response to finding a match causing a second object of the application to perform operations defined by the instruction set corresponding to the matched digital picture.[26] For example, if a second object detects a digital picture (e.g., picture of virtual debris or virtual people in the road) and the system has a similar digital picture representation that matches with the detected picture, then the second object performs an instruction set (e.g., braking). The corresponding claimed features, as a combination, are inventive and novel and were not well-understood, routine or conventional as of the priority date of the ’585 patent.

66. Referring to Figure 35 of the ’585 patent, for example, object 180b (e.g., a vehicle) includes, in its knowledgebase, associated digital pictures 525 of the objects surrounding in a 3D virtual world 18b. ’585 patent at 166:38-167:67; Fig. 35; *see also* ’134 patent at 163:52-166:46; Fig. 36. If the system detects a match with a previously learned digital picture 525, the object 180b performs autonomous operation using previously learned instructions 526 correlated with the previously learned digital picture. ’585 patent at 166:38-167:67; Fig. 35; *see also* ’134 patent at 163:52-166:46; Fig. 36. In my opinion, the corresponding claimed features were not well-understood, routine or conventional as of the priority date of the Simulation Patents.

[24] ’585 patent at cl. 1 (“receiving or generating a new one or more digital pictures that depict at least a portion of a surrounding of: the first object of the first application program, a second object of the first application program, or a first object of a second application program”)
[25] ’585 patent at cl. 1 (“determining the first one or more instruction sets for operating the first object of the first application program based on at least partial match between the new one or more digital pictures and the first one or more digital pictures”)
[26] ’585 Patent at cl. 1 (“at least in response to the determining, executing the first one or more instruction sets for operating the first object of the first application program, wherein the first object of the first application program, the second object of the first application program, or the first object of the second application program autonomously performs one or more operations defined by the first one or more instruction sets for operating the first object of the first application program.”)

67. Given that the memory stores a first correlation (picture or object representation correlated with instructions learned by at least partially operating the first device) acquired by a first avatar, when a second avatar, for example, encounters a partially matching picture or object representation, the second avatar can autonomously perform an operation defined by the previously learned instruction that is correlated with the picture or object representation from the first avatar. The ability to cause one avatar to autonomously perform instruction(s) based on comparing a picture or object representation encountered by that device with a picture or object representation encountered by another or the same device was not known or done in the prior art.

68. The above-described claimed inventions, however, provided a solution to this problem. By allowing one avatar's response (e.g., driver's response when driving a vehicle) to a circumstance to train other avatars (e.g., vehicles in a simulation) that encounter similar circumstances, the claimed knowledge base, together with the claimed anticipating and executing, allows a group of devices (e.g., a fleet of vehicles) to continuously and dynamically learn to respond autonomously to new and unforeseen circumstances, which occur in the simulation. Over time, nearly every circumstance can be accounted for and subsequently applied autonomously by all devices (e.g., the fleet of vehicles). The claimed features thus provide a specific and improved way for an avatar—and by proxy a real life vehicle—to respond autonomously to detected circumstances.

69. With the innovations in the Autonomous Vehicle Simulation Patents, autonomous device operation using a knowledgebase (e.g., a neural network that includes, for example, a combination of object recognition and correlated previously learned instructions) that is trained by a fleet of vehicles became possible. *See, e.g.*, '134 Patent at 103:8-104:64; '585 Patent at 105:19-107:4. The corresponding claimed features were not well-understood, routine or

conventional as of the priority date of the Simulation Patents. Moreover, the claimed features in the Simulation Patents cannot be performed as mental steps by a human, nor do they represent the application of a generic computer to any well-known method of organizing human behavior.

70. The claims of the Autonomous Vehicle Simulation Patents are directed to non-abstract ideas in that they provide technical solutions to at least the technical problems described above. For instance, claim 1 of each Simulation Patents, as a whole, is inventive and novel, as are at least the above-identified claim limitations. As of the priority dates of the Simulation Patents, the concept of causing a virtual device to perform autonomous operation based on: (i) pictures/representations of that device's circumstances and (ii) pictures/representations and instructions learned from that device or another device was not well-understood, routine, or conventional. These features provide a vast improvement over the prior art.

71. The claimed features unlocked the next level of autonomous driving. While fleet learning from real-world pictures and object representations can, in general, work, there were still inhibiting limitations such as scale and uniqueness. Regarding scale, the fleet is limited in its learning ability by the number of autonomous vehicles on the road that are conveying data back into the system. Similarly, regarding uniqueness, the fleet is limited in what it can learn based on the monitored autonomous vehicles experiences. The Simulation Patents address these issues by taking the concrete ideas found in the Object Representation and Digital Picture Patents and importing them into proper simulations. As discussed in the prosecution of both Simulation Patents, this was a completely novel idea as the prior art does not teach the details of object representations, avatars, determining instruction sets, and causing an avatar to perform operations as recited in the claims. By allowing the entire fleet to train the system, as discussed earlier, and allowing appropriate simulations of driving reactions and situations including those

situations that may represent improbable scenarios and edge cases, much more precise and appropriate reactions to driving circumstances became possible. This is because necessary but less probable or seldom occurring scenarios can be distilled into a set of circumstances/pictures and corresponding instructions that can be efficiently performed autonomously.

72. The claims of the Simulation Patents recite one or more inventive concepts rooted in computerized technology that overcome technical problems in that field. A person of ordinary skill in the art reading the Simulation Patents and their claims would understand that the Simulation Patents' disclosures and claims are drawn to solving specific technical problems arising in simulating artificially intelligent and autonomous devices and systems. Accordingly, each claim of the Simulation Patents recites a combination of elements sufficient to ensure that the claim in practice amounts to significantly more than a patent claiming an abstract concept. Further, the claimed improvements over the prior art are concrete and improve the capabilities of existing autonomous and AI simulation systems.

73. A person of ordinary skill in the art reviewing the specification of the Simulation Patents would understand that the inventor had possession of the claimed subject matter and would know how to practice the claimed invention without undue experimentation.

VII. CONCLUSION

74. I declare under penalty of perjury under the laws of the United States that the foregoing is true and correct.

Executed on September 14, 2023.

By:______________________________

Eli Saber, Ph.D.

APPENDIX A

Prof. Eli Saber

Mobile: (585) 727-3126, **Email:** esaber228@gmail.com, **HTTP:** http://people.rit.edu/esseee

EDUCATION

Ph.D., Electrical Engineering, University of Rochester, Rochester, New York (March 1996)
Concentration: Signal/Image/Video Processing, Pattern Recognition, and Computer Vision.
Dissertation: Automatic image annotation and query-by-example using color, shape and texture information.

M.S., Electrical Engineering, University of Rochester, Rochester, New York (May 1992)
Concentration: Signal/Image/Video Processing, Pattern Recognition, Computer Vision, Communications.

B.S., Electrical and Computer Engineering, Summa Cum Laude, State University of New York at Buffalo, Buffalo, New York (May 1988).
Concentration: Computers, Microprocessors, Communications, Instrumentation.

A.S., Engineering Science, Mohawk Valley Community College, Utica, New York (May 1986)

RESEARCH INTERESTS

- Image and Video Processing for Multimedia, Military & Biomedical Applications.
- Computer Vision and Three-Dimensional Scene Reconstruction.
- Color Image Processing for Printing and Multimedia Applications.

CAREER MAJOR HIGHLIGHTS

- **Academic Experience:**
  - **Teaching:** Taught/continue to teach undergraduate and graduate courses in the signal/image/video processing, computer vision, controls and communications focus areas with excellent feedback from students from 1996 to Present. Founded new coursework in Digital Video Processing and Pattern Recognition.
  - **Research:** Conducted research on multiple acquired government and industry grants in the areas of: 1) image/video processing for multimedia, military and biomedical applications, 2) deep learning/artificial intelligence for object detection and tracking purposes, 3) three-dimensional scene reconstruction for remote sensing, and 4) color image processing for printing and multimedia applications. Graduated several PhD and MS students. Currently co-advising 2 PhD students and 2 MS students.
  - **External Funding:** Acquired external funding as PI in excess of $3.5 Million and as PI/Co-PI in excess of $5 Million over the period of September 2004 to present from government agencies and industry partners.
  - **Publications:** 1 Book, 2 Special Issues, 38 Journal, 98 Conference and 11 Patents/Patent Publications.
  - **Service:** Served/chaired several department, college and university level committees. Chaired the Graduate Program at the EME department for a period of five years. Served on the academic senate for a period of 9 years and on the Academic Senate Executive Committee for a period of 3 years.
- **Industrial Experience:** Worked for Xerox Corporation from 1988 until 2004 in a variety of engineering, managerial and scientific positions ending as Product Development Scientist and Manager.
- **Consulting/Litigation Experience:** Served as an expert in several patent and non-patent cases representing various companies such as Lyft, Genetec, Interactive Digital Solutions, Hisense, Perfect Corporation, Corephotonics, Wells Fargo, Netflix, Blackberry, 3Shape, Nikon, Hewlett-Packard, Qomo Hitevission, Sony, and

Canon. Provided declarations in support of multiple Inter Partes Reviews (IPR), expert opinions and reports with regards to validity/invalidity, infringement/non-infringement and domestic industry, tutorial at Markman hearing and testimony in court proceedings.

- **Professional Activities:** Served on several IEEE and SPIE conferences in various chair positions including Finance, Tutorials, Plenaries and Technical Program. Served as general chair for two conferences.
- **Honors & Awards:** RIT Trustees Award (2012), KGCOE Scholarship Award (2012), EME Gleason Professor (2011-2013), and PI Millionaire (2011).

ACADEMIC EXPERIENCE

Professor (Associate 2004 – 2010, Full 2010 – Present), Department of Electrical and Microelectronic Engineering (EME), Kate Gleason College of Engineering (KGCOE), Rochester Institute of Technology.
Extended Faculty (2004 – Present), Center for Imaging Science (CIS), Rochester Institute of Technology.
Director of the Image, Video and Computer Vision Laboratory (2004 – Present)
Graduate Program Director (2010 – 2014)

- **Teaching**: Responsible for teaching undergraduate & graduate courses in Digital Signal Processing, Digital Image Processing, Digital Video Processing, Engineering Analysis, Advanced Engineering Mathematics, Random Signal & Noise, Pattern Recognition, Communication Systems, Digital Data Communications, Computer Vision, Modern Control Theory, Linear Systems, and Matrix Methods. Typically teaching two courses per semester as required by department. Have taught three per semester on occasion to fulfill department needs.
- **Research:** Director of the Image, Video and Computer Vision Laboratory. Currently co-advising 2 PhD and 2 MS students on deep learning/machine learning/artificial intelligence applications for object detection and tracking. Advised several PhD and MS students in image/video segmentation, hierarchical image decomposition, video mosaicking/3-dimensional scene reconstruction, image/video understanding, object tracking/recognition. All students are/were funded under various government or corporate grants.
- **Funding:** Acquired funding as PI in excess of $3.5 Million and as PI/Co-PI in excess of $5 Million over the period of September 2004 – Present from various government agencies and industrial partners.
- **Service (Department, College and University):**
  - Advised/currently advising several undergraduate and graduate students on curriculum issues.
  - Member of the Academic Senate Resource Allocation and Budget Committee, the University Compensation Committee and of the Kate Gleason College of Engineering Promotion Committee.
  - Former Member (KGCOE representative) of the Academic Senate (2007 – 2013 and 2014-2017) and of the Academic Senate Executive Committee (2009-2010, 2016-2017).
  - Served as Chair of the RIT Vision 2025 committee per request from RIT President Dr William Destler. Committee was commissioned, over the summer of 2009, to review 80+ ideas submitted from all colleges and provide a recommendation to upper administration.
  - Former Member and Chair of KCGOE graduate committee.
  - Former Member of Graduate Council.
  - Served on several committees for curriculum development, recognition, and faculty search.
- **Professional Activities:**
  - Senior member of the Institute of Electrical and Electronic Engineers (IEEE) society.
  - Former Member of the IEEE Industry Technical Committee on DSP.
  - Former Member of the IEEE Image and Multidimensional Digital Signal Processing (IMDSP) technical Committee.
  - Former Area Editor for the Journal of Electronic Imaging.
  - Former member of the Imaging Science and technology (IS&T) society.
  - International Conference on Image Processing (ICIP) 2002 Finance Chair, ICIP 2007 and ICIP 2009 Tutorial Chair, ICIP 2012 General Chair, ICASSP 2017 Technical Program Chair, ICIP 2021 Plenary Chair.
  - Co-founder and General Chair of the Video Surveillance and Transportation Imaging Conference within the Electronic Imaging Symposium.
- **Honors and Awards:**

- Awarded the Prestigious Trustees Scholarship award – the highest award at RIT with regards to research recognition
- Elected in 2011 as PI Millionaire.
- Awarded the EME Gleason Professor for 3 years (2011-2013).
- Elected as the Kate Gleason College of Engineering Scholarship award winner.

Adjunct Faculty Member, Dept. of Electrical & Computer Engineering, University of Rochester. (09/96-07/04)

- Taught undergraduate & graduate courses in Digital Signal Processing, Digital Image Processing, Pattern Recognition/Advanced Image Processing, Detection/Estimation Theory, and Analog & Digital Communications.
- Advised and graduated 1 Ph.D. student in the areas of "Image Understanding" and "Database Content Indexing".
- Co-advised 1 Ph.D. student in the areas of "Watermarking".
- Advised and graduated 1 MS student in the area of "Color Rendering" and "Printer Characterization".
- Served as a committee member on several doctoral dissertations.
- Served on PhD qualifying examinations for the Signal Processing and Communications Concentration.
- Sought and captured funding from the National Science Foundation for the development of an intelligent image database system. Proposal funded for 4 years under NSF Grant IIS – 9820721
- Sought and captured industrial funding from Xerox Corporation for Printer Color Characterization & Digital Front End Object Oriented Rendering.

Adjunct Faculty Member, Dept. of Electrical Engineering, Rochester Institute of Technology. (03/98-07/04)

- Taught undergraduate and graduate courses in Pattern Recognition, Digital Video Processing, Random Signal & Noise, Image and Video Compression, and Communications.
- Advised and graduated a master student in the area of "Texture Classification".

INDUSTRIAL EXPERIENCE

Product Development Scientist & Manager, Print Engine Development Unit, Xerox Corporation. (10/98-08/04)
Major responsibilities included:

- Lead the Image Science, Analysis and Evaluation area (12-15 direct reports and ~$2 Million budget).
- Lead the development of highlight color specifications for the Sorrento print engine.
- Lead the development of color characterization algorithms for the iGen3 print engine.
- Lead the image quality integration of two color front end for the iGen3 Product.
- Lead the development of ROS and LED based imaging systems and image path architectures for upcoming highlight & full color products.
- Lead the development of xerographic hardware/algorithms & imaging systems for the DP92C highlight color product. (Product launched 9/30/99 and follow-on launched 4/20/00)
- Lead the research and development of image quality metrics for various product platforms and their dissemination throughout the Print Engine Development Unit and Xerox Corporation.
- Collaborate with the Department of Electrical & Computer Engineering (Univ. of Rochester) & the Center for Electronic Imaging Systems.

Advanced Development Scientist and Manager, Print Cartridge Delivery Unit, Xerox Corporation. (2/97 – 9/98)
Major responsibilities included:

- Establish the Advanced Design Laboratory (an imaging/xerographics lab) and provide technical and managerial leadership for the Electrical, Imaging and Xerographics Dept.
- Perform image processing and xerographic hardware/software design and development for low/mid volume color copiers and printers for current and future programs.
- Perform technology development, modeling, and product design for upcoming Xerox color products, specifically image on paper and image on belt products..
- Lead the development of the xerographic module for a color intermediate belt transfer product with direct technical and management responsibilities.

- Collaborate with the Department of Electrical & Computer Engineering (Univ. of Rochester) & the Center for Electronic Imaging Systems.

Research and Development Scientist, Production Systems Group, Xerox Corporation. (1/96-1/97)
Major responsibilities included:
- Lead the design and development of color characterization/management and image quality algorithms and specifications for digital front ends destined to drive high quality, high speed color print engines.
- Integrate color management & image processing algorithms into the Raster Image Processing module.
- Participate in the design and development of a high speed raster image processing architecture.
- Benchmark developed algorithms against existing products & systems both internally and externally.
- Collaborate with the Department of Electrical & Computer Engineering of the University of Rochester and the Center for Electronic Imaging Systems.

Research and Development Engineer, Corporate Research, Xerox Corporation (8/93-12/95) & Department of Electrical & Computer Engineering, University of Rochester. (1/95-12/95). Major responsibilities included:
- Design and develop query by image content and query by example image annotation algorithms utilizing color, shape, texture and motion cues. System is able to perform query by keywords, color, shape, texture, and/or a combination of the above cues.
- Design and develop intelligent image segmentation algorithms. These algorithms are currently utilized in the query by image content and query by example systems described above.
- Design and develop face detection and facial feature extraction approaches.
- Design and develop color characterization/calibration and image quality algorithms for Digital Front Ends aimed at driving high speed / high quality print engines.

(Note: Image annotation/content analysis research was done in conjunction with the Department of Electrical & Computer Engineering and Center for Electronic Imaging Systems leading to the Ph.D.)

Electronic, Computer and Instrumentation Engineer, New Toner/Developer Facility Engineering, Xerox Corporation. (6/88-7/93). Major responsibilities included:
- Provide design, development, installation, startup, and training for multiple toner production facilities.
- Provide development and implementation of control system database, software and displays for several systems.
- Evaluate vendor supplied electrical specifications and drawings.
- Manage and coordinate the efforts of technicians, electrical support, construction crew, and industrial workforce during the design, construction, startup, and implementation phases.
- Supervise and complete a number of upgrade projects for toner & photoreceptor production including software development, preparation of electrical design, procurement of necessary equipment and parts, supervision of technicians, contractors and industrial workforce, and scheduling of construction.

During this time, I gained extensive experience in the following systems: Fisher distributive control, unit operation controller, Provue console, Acrison material handling, Werner and Pfleiderer extrusion, Alpine air grinding, Majac/Micropul centrifugal classifiers, dry/wet material screening, Waeschle and others bulk powder storage and pneumatic convey, Ingersoll Rand and Joy compressed air equipment, Statistical process control.

CONSULTING EXPERIENCE

Expert on Legal Cases: Served as expert in multiple patent cases representing various companies such as Lyft, Genetec, Interactive Digital Solutions, Hisense, Perfect Corporation, Corephotonics, Wells Fargo, Netflix, Blackberry, 3Shape, Nikon, Hewlett-Packard, Qomo Hitevission, Sony, and Canon.. Provided: 1) declarations in support of multiple Inter Partes Reviews (IPR); 2) expert opinions and reports with regards to validity/invalidity, infringement/non-infringement and domestic industry; 3) tutorial at Markman hearing and 4) testimony in multiple court proceedings.

Summary of Experience:

- IPR/Declarations: Provided declarations in support of 19 IPR proceedings.
- Depositions: Provided 11 Depositions.
- Court Testimony: Testified in International Trade Commission Court in Sept. 2018 and Oct. 2019.
- Expert Opinions: Provided multiple expert opinions and reports in support of validity/invalidity, infringement/non-infringement, domestic industry and rebuttals.
- Claim Construction: Provided reports in support of claim construction.
- Hearings: Attended one Markman hearing and provided a brief technology tutorial.

Industrial Training: Developed and conducted industrial training at Xerox Corporation for engineering personnel over three separate summer periods in the areas of signal and image processing, color engineering, and control systems with a distinct focus on digital front ends and print engines. I have also provided pattern recognition/shape matching expertise for Leica corporation.

BOOKS

1. S. Dianat and E. Saber, "Advanced Linear Algebra for Engineers with MATLAB", CRC press, February 2009.

SPECIAL ISSUES

1. H. J. Trussell, E. Saber and M. Vrhel, "Color Image Processing", IEEE SP magazine, January 2005.
2. R. Loce and E. Saber, "Video Surveillance and Transportation Imaging", Journal of Electronic Imaging, 22(4), Dec. 2013

PEER-REVIEWED JOURNAL PUBLICATIONS

1. M. Sharma, M. Dhararaj, S. Karnam, D. Chachlakis, R. Ptucha, P. Markopoulos and E. Saber, "YOLOrs: Object Detection in Multimodal Remote Sensing Imagery", IEEE Journal of Selected Topics in Applied Observations and Remote Sensing, Vol. 14, 2021.
2. S. Piramanayagam, E. Saber, and N. Cahill, "Gradient Driven Unsupervised Video Segmentation using Deep Learning Techniques", Journal of Electronic Imaging, 29 (1), 013019, 2020
3. U. Gewali, S. Monteiro and E. Saber, "Spectral Super-Resolution with Optimized Bands", Remote Sensing, 11(14), 2019
4. U. Gewali, S. Monteiro and E. Saber, "Gaussian Processes for Vegetation Parameter Estimation from Hyperspectral Data with Limited Ground Truth", Journal of Remote Sensing, 11(13), 2019.
5. Y. Liang, P. Markopoulos and E. Saber, "Spatial-Spectral Segmentation of Hyperspectral Images for Subpixel Target Detection", Journal of Applied Remote Sensing, 13(3), 2019.
6. Y. Liu, S. Piramanayagam, S. Monteiro, and E. Saber. "Semantic segmentation of multisensor remote sensing imagery with deep ConvNets and higher-order conditional random fields." Journal of Applied Remote Sensing 13, no. 1 (2019): 016501
7. U. Gewali, S. Monteiro and E. Saber, "Hyperspectral Image Analysis using Machine Learning: A Survey", arXiv preprint, 2018.
8. S. Piramanayagam, E. Saber, W. Schwartzkopf, F.W. Koehler, "Supervised Classification of Multisensor Remotely Sensed Images using a Deep Learning Framework", Remote Sensing Journal, 10 (9), 2018.
9. S. R. Vantaram, Y. Hu, E. Saber and S. Dianat, "Synthesis of Intensity Gradient and Texture Information for Efficient Three-Dimensional Segmentation of Medical Volumes", Journal of Medical Imaging 2, no. 2 (2015): 024003-024003.

10. S. R. Vantaram, S. Piramanayagam, E. Saber and D. Messinger, "Automatic Spatial Segmentation of Multi/Hyperspectral Imagery by Fusion of Spectral-Gradient-Textural Attributes", Journal of Applied and Remote Sensing, Vol. 9, No. 1, pp. 095086 (1-37), 2015.
11. S. R. Vantaram and E. Saber, "A Survey of Contemporary Trends in Color Image Segmentation", Journal of Electronic Imaging, 21(4), 040901, Oct-Dec 2012.
12. M. S. Erkilinc, M. Jaber, E. Saber, "Text, Photo and Line Extraction in Scanned Documents", Journal of Electronic Imaging, Vol. 21, 033006, July 2012.
13. T. Keane, E. Saber, H. Rhody, A. Savakis and J. Raj, "Practical Image Registration Concerns Overcome by the Weighted and Filtered Mutual Information Metric", Journal of Electronic Imaging, Vol. 21(2), 023029, June 2012.
14. P. Gurram, E. Saber and H. Rhody, "Semi-automated System for three-dimensional Modeling of Buildings from Aerial Video", Journal of Electronic Imaging, Vol. 21(1), 013007, Jan-Mar 2012.
15. M. Jaber and E. Saber, "Probabilistic Approach for Extracting Regions of Interest in Digital Images", Journal of Electronic Imaging, Vol. 19, No. 2, April - June 2010.
16. X. Fan, H. Rhody and E. Saber, "A Spatial Feature Enhanced MMI Algorithm for Multimodal Airborne Image Registration", IEEE Transaction on Geoscience and Remote Sensing, Vol. 48, Issue 6, pp. 2580 – 2589, 2010.
17. P. Gurram, E. Saber and H. Rhody, "A Segment-Based Mesh design for Building Parallel-Perspective Stereo Mosaic", IEEE Transactions on Geoscience and Remote Sensing, Vol. 48, No. 3, March 2010.
18. S. Vantaram, E. Saber, S. Dianat, M. Shaw and R. Bashkar, "Multiresolution Adaptive and Progressive Gradient-based color image Segmentation", Journal of Electronic Imaging, Volume 19, Number 1, pp. 1-21, January-March 2010.
19. L. Garcia, E. Saber, S. Vantaram, V. Amuso, M. Shaw and R. Bhaskar, "Automatic Image Segmentation by Dynamic Region Growth and Multi-resolution Merging", IEEE Transactions on Image Processing, Vol. 18, No. 10, Oct. 2009.
20. H. Santos, E. Saber, and W. Wu, "Streak Detection in Mottled and Noisy Images", Journal of Electronic Imaging, Vol. 16, No. 4, 2007.
21. O. Ugbeme, E. Saber and W. Wu, "An Automated Algorithm for the Identification of Artifacts in Mottled and Noisy Images", Journal of Electronic Imaging, Vol. 16, No. 3, 2007.
22. V. Misic, V. Sampath, Y. Yu and E. Saber, "Prostate Boundary Detection and Volume Estimation Using TRUS Images for Brachytherapy Applications", International Journal of Computer Assisted Radiology and Surgery, Vol. 2, No. 2, pp. 87-98, August 2007.
23. A. Ononye, A. Vodacek and E. Saber, "Towards Automatic Extraction of Fire Line Parameters from Multispectral Infrared Images", Journal of Remote Sensing of Environment, Vol. 108, pp. 179 – 188, 2007.
24. E. Saber, S. Dianat and L. Mestha, "DSP utilization in Digital Color Printing", IEEE SP Magazine, July 2005.
25. E. Saber, Y. Xu, and A. M. Tekalp, "Partial Shape Recognition by sub-matrix matching for partial matching guided image labeling", Pattern Recognition, Vol. 38, pp. 1560 – 1573, 2005.
26. M. Celik, G. Sharma, A. M. Tekalp, E. Saber, "Lossless Generalized-LSB Data Embedding", IEEE Trans. on Image Processing, Vol. 14, No. 2, pp. 253 – 266, February 2005.
27. M. Vrhel, E. Saber, and H. J. Trussell, "Color Image Generation and Display Technologies: An overview of methods, devices, and research", IEEE Signal Processing Magazine, January 2005.
28. H. J. Trussell, E. Saber and M. Vrhel, "Color Image Processing: Basics and Special Issue Overview", IEEE Signal Processing Magazine, January 2005.
29. Y. Xu, E. Saber, and A. M. Tekalp, "Dynamic Learning from Multiple Examples for Semantic Object Segmentation and Search", Computer Vision and Image Understanding, Vol. 95, No. 3, pp. 334-353, Sept 2004.
30. Y. Xu, P. Duygulu, E. Saber, A. M. Tekalp, and F. T. Yarman-Vural, "Object-Based Image Labeling through Learning by Example and Multi-Level Segmentation", Pattern Recognition, Vol. 36 (6), pp. 1407-1423, June 2003.
31. Y. Xu, E. Saber, and A. M. Tekalp, "Object Segmentation and Labeling by Learning from Examples", IEEE Transactions on Image Processing, Vol. 12, No. 6, June 2003.
32. M. Celik, G. Sharma, E. Saber, and A. M. Tekalp, "Hierarchical watermarking for secure image authentication with localization", IEEE Trans. on Image Processing, vol. 11, no. 6, June 2002.
33. M. Xia, E. Saber, G. Sharma, and A. M. Tekalp, "End-to-End Color Calibration by Total Least Squares Regression", IEEE Transactions on Image Processing, Vol. 8, No. 5, May 1999.

34. E. Saber and A. M. Tekalp, "Facial Pattern Detection and Eye Localization using Color, Shape and Symmetry-Based Cost Functions", Pattern Recognition letters, Vol. 19, 1998.
35. E. Saber and A. M. Tekalp, "Integration of Color, Shape and Texture for Automatic Image Classification, Annotation and Retrieval", Journal of Electronic Imaging, Vol. 7, No. 3, July 1998.
36. E. Saber and A. M. Tekalp "Region-Based Affine Shape Matching for Automatic Image Annotation and Query-by-Example", Journal of Visual Communication and Image Representation, March 1997.
37. E. Saber, A. M. Tekalp, and G. Bozdagi, "Fusion of Color and Edge Information for Improved Segmentation and Edge Linking,", Image and Vision Computing, Vol. 15, 1997.
38. E. Saber, A. M. Tekalp, R. Eschbach and K. Knox, "Automatic Image Annotation using Color Classification", Graphical Models and Image Processing, Volume 58, Number 2, March 1996.

CONFERENCE & WORKSHOP PUBLICATIONS

1. M. Sharma, P. P. Markopoulos, E. Saber, M. S. Asif, and A. Prater-Bennette, "Convolutional Auto-Encoder with Tensor-Train Factorization," Proc. International Conference on Computer Vision, (ICCV 2021), RLS-CV workshop.
2. M. Sharma, P. P. Markopoulos, and E. Saber, "YOLOrs-LITE: A Lightweight CNN for Real-time Object Detection in Remote Sensing," Proc. IEEE International Geoscience and Remote Sensing Symposium (IEEE IGARSS), Brussels, Belgium, July 2021.
3. M. Dhanaraj, M. Sharma, T. Sarkar, S. Karnam, D. Chachlakis, R. Ptucha, P. Markopoulos and E. Saber, "Vehicle Detection from Multi-modal Aeriel Imagery using YOLOv3 with Mid-level Fusion", Big Data II: Learning, Analytics, and Applications, SPIE, April 2020.
4. Y. Liu, S. Monteiro and E. Saber, "Dense Semantic Labeling of Very High Resolution Aerial Imagery and LIDAR with Fully Convolutional Neural Netwroks and Higher Order CRFs", CVPR, 2017
5. Y. Liu, S. Piramanayagan, S. Monteiro and E. Saber, "Semantic segmentation of remote sensing data using Gaussian processes and higher order CFRs", IGARSS, Fort Worth, TX, July 2017.
6. Y. Liang, S. T. Monteiro, and E. Saber, "Gaussian processes for object detection in high resolution remote sensing images", IEEE International Conference on Machine Learning and Applications (ICMLA 2016), Anaheim, CA, December 2016.
7. S. Piramanayagam, W. Schwartzkopf, F.W. Koehler, E. Saber, "Classification of remote sensed images using random forests and deep learning framework", Proc. SPIE 10004, Image and Signal Processing for Remote Sensing XXII, 100040L, October 2016;
8. Y. Liang, S. T. Monteiro, and E. Saber, "Transfer learning for high resolution aerial image classification" ,to appear in IEEE Applied Imagery Pattern Recognition Workshop (AIPR 2016), Washington, D.C., October 2016.
9. Y. Liang, P. P. Markopoulos, and E. Saber, "Subpixel target detection in hyperspectral images with local matched filtering in SLIC superpixels", IEEE Workshop on Hyperspectral Image and Signal Processing: Evolutions in Remote Sensing (WHISPERS 2016), Los Angeles, CA, August 2016.
10. Y. Hu, S. Monteiro and E. Saber, "Super Pixel Based Classification using Conditional Random Fields for Hyperspectral Images", ICIP 2016, Phoenix, AZ.
11. O. de Lima, S. Janakiraman, E. Saber, D. C. Day, M. Shaw, P. Bauer. R. S. Twede, and P. Lea, "Signature Line Detection in Scanned Documents", ICIP 2016, Phoenix, AZ.
12. Y. Liang, P.P. Markopoulos, and E. Saber", Subpixel Target Detection in Hyperspectral Images from Superpixel Background Statistics," IGARSS, Beijing, China, July 2016.
13. Y. Liu, S.T. Monteiro, and E. Saber. "Vehicle detection from aerial color imagery and airborne LiDAR data", IGARSS, Beijing, China, 2016.
14. Y. Wang, J. Mathew, E. Saber, D. Larson, P. Bauer, G. Kerby and J. Wagner. "Scanned Document Enhancement Based on Fast Text Detection," International Conference on Acoustics, Speech and Signal Processing, Shanghai, China, 2016
15. Y. Liu, S. Monteiro, and E. Saber, "An Approach for Combining Airborne LiDAR and High Resolution Aerial Color Imagery using Gaussian Processes", Proc. SPIE 9643, Image and Signal Processing for Remote Sensing XXI, Oct. 2015.
16. Y. Liang, N. Cahill, E. Saber and D. Messinger, "A Game-Theoretic Tree Matching Approach for Object Detection in High Resolution Remotely Sensed Images", Proc. SPIE 9643, Image and Signal Processing for Remote Sensing XXI, Oct. 2015.

17. Y. Hu, S. T. Monteiro and E. Saber, “Comparing Inference Methods for Conditional Random Fields for Hyperspectral Image Classification”, *Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing*, Tokyo, Japan, 2015
18. Y. Hu, N. Cahill, S. Monteiro, E. Saber and D. Messinger, “Dimensionality Reduction for Hyperspectral Imagery Classification in Conditional Random Fields”, Proc. SPIE 9643, Image and Signal Processing for Remote Sensing XXI, Oct. 2015.
19. S. Piramanayagam, P. J. Cutler, W. Schwartzkopf, F.W. Koehler, E. Saber, “Application of gradient based image segmentation to SAR imagery”, IGARSS, Milan, Italy, July 2015.
20. S. Piramanayagam, E. Saber, N. D. Cahill, and D. Messinger, “Shot Boundary Detection and Label Propagation for Spatio-Temporal Video Segmentation”, SPIE/IS&T: Electronic Imaging Symposium, San Francisco, CA, Feb, 2015.
21. M. Yousefhussien, R. Easton, R. Ptucha, M. Shaw, B. Bradburn, J. Wagner, D. Larson and E. Saber, “Flatbed Scanner Simulation to Analyze the Effect of Detector’s Size on Color Artifacts”, SPIE/IS&T: Electronic Imaging Symposium, San Francisco, CA, Feb. 2015.
22. K. Shah, E. Saber and K. Verrier, "Improved Metrology of Implant Lines on Static Images of Textured Silicon Wafers using Line Integral Method”, SPIE/IS&T: Electronic Imaging Symposium, San Francisco, CA, Feb. 2015.
23. Y. Hu, E. Saber, S. Monteiro, N. Cahill and D. Messinger, “Classification of Hyperspectral Images Based on Conditional Random Fields”, SPIE/IS&T: Electronic Imaging Symposium, San Francisco, CA, Feb. 2015.
24. R. Kothari, E. Saber, M. Nelson, M. Stauffer and D. Bohan, “Image Enhancement for Low Resolution Display Panels”, SPIE/IS&T: Electronic Imaging Symposium, San Francisco, CA, Feb. 2015.
25. Y. Liang, A. H. Syed, N. Cahill, E. Saber, and D. Messinger, “Application of Tree Matching Techniques to High Resolution Remotely Sensed Images toward Object Detection”, GEOBIA, May 2014.
26. Y. Hu, A. H. Syed, E. Saber, N. Cahill, and D. Messinger, “Dynamic scale-space representation based on a MRF region merging model”, GEOBIA, May 2014
27. Y. Liu, M. Helguerra and E. Saber, “Measurement of blood flow velocity in vivo video sequences with motion estimation methods”, SPIE Medical Imaging, San Diego, 2014.
28. Y. Wang, O. F. de Lima, and E. Saber, “Improved edge directed Super resolution with hardware realization for surveillance, transportation, and multimedia applications”, SPIE/IS&T: Electronic Imaging Symposium, San Francisco, CA, Feb. 2014.
29. J. Whitesell, D. Patru, E. Saber, G. Roylance and B. Larson, “Design for Implementation of Color Image Processing Algorithms”, SPIE/IS&T: Electronic Imaging Symposium, San Francisco, CA, Feb. 2014
30. R. Toukatly, D. Patru, E. Saber, E. Peskin, G. Roylance and B. Larson, “Performance Analysis of a Color Space Conversion Engine Implemented using Dynamic Partial Reconfiguration”, SPIE/IS&T: Electronic Imaging Symposium, San Francisco, CA, Feb. 2013.
31. O. De Lima, S. R. Vantaram, S. Piramanayagam, E. Saber and K. Bengtson, “An Edge Directed Super Resolution Technique for Surveillance and Printing Applications”, SPIE/IS&T: Electronic Imaging Symposium, San Francisco, CA, Feb. 2013.
32. M. YousefHussien, K. Garvin, D. Dalecki, E. Saber and M. Helguera, “Three-Dimensional Volume Analysis of Vasculature in Engineered Tissues”, SPIE/IS&T: Electronic Imaging Symposium, San Francisco, CA, Feb. 2013
33. A. Mykyta, D. Patru, E. Saber. G. Roylance and B. Larson, “Reconfigurable Framework for High-Bandwidth Stream-Oriented Data Processing”, IEEE International SoC Conference (SOCC), Buffalo, NY, Sept. 2012.
34. S. R. Vantaram, and E. Saber, “Unsupervised Video Segmentation by Dynamic Volume Growing and Multivariate Volume Merging using Color-Texture-Gradient Features”, *IEEE International Conference on Image Processing, Orlando, Florida, September 2012*.
35. A. H. Syed, E. Saber and D. Messinger, “Encoding of Topological Information in Multi-Scale Remotely Sensed Data: Applications to Segmentation and Object-Based Image Analysis”, International Conference on geographic Object-Based Image Analysis (GEOBIA), Rio de Janeiro, May 2012. (Best paper Award)
36. S. Piramanayagam, E. Saber and D. Heavner, “Measurement of Glucose by Image Processing of Thin Film Slides”, SPIE: Medical Imaging, San Diego, CA, 2012.
37. Y. Hu, E. Saber, S. Dianat, S. R. Vantaram and V. Abhyankar, “An Automatic Approach for 3D Registration of CT Scans”, SPIE/IS&T: Electronic Imaging Symposium, San Francisco, CA, Jan. 2012
38. S. R. Vantaram, E. Saber, S. Dianat, Y. Hu and V. Abhyankar, “Semi-Automatic 3-D Segmentation of Computed Tomographic Imagery by Iterative Gradient-Driven Volume Growing”, *IEEE International Conference on Image Processing, Brussels, Belgium, September 2011.*

39. S. R. Vantaram**,** E. Saber and D. Messinger, "An Intensity-Gradient-Texture Guided Methodology for Automatic Spatial Segmentation of Remotely Sensed Multi/Hyperspectral imagery", *IEEE International Conference on Image Processing, Brussels, Belgium, September 2011.*
40. S. R. Vantaram, and E. Saber, "A Method for Improved Localization of Edges in multi/hyperspectral Imagery", Proceedings of the SPIE Optical Engineering + Applications Conf., San Diego, CA, Aug. 2011.
41. M. Jaber, M. S. Bailly, Y. Wang, and E. Saber, "An image-set for identifying multiple regions/levels of interest in digital images," *Proceedings of the SPIE Optical Engineering + Applications Conference*, San Diego, CA, Aug. 2011.
42. M. S. Erkilinc, M. Jaber, E. Saber, P. Bauer and D. Depalov, "Page layout analysis and classification in complex scanned documents," *Proceedings of the SPIE Optical Engineering + Applications Conference*, San Diego, CA, Aug. 2011.
43. S. R. Vantaram, and E. Saber, "An Adaptive Bayesian Clustering and Multivariate Region Merging-based Technique for Efficient Segmentation of Color Images", IEEE International Conference on Acoustics, Speech and Signal Processing, Prague, Czech Republic, May 2011.
44. A. H. Syed, E. Saber and D. Messinger, "Scale-space Representation of Remote Sensing Images using an Object-Oriented Approach", SPIE Defense, Security, and Sensing, Orlando, Florida, Apr 2011.
45. M. Jaber and E. Saber, "Image Understanding Algorithm for Segmentation Evaluation and Region-of-Interest Identification using Bayesian Networks", SPIE Defense, Security, and Sensing, Orlando, Florida, Apr 2011.
46. T. Keane, E. Saber, H. Rhody, A. Savakis and J. Raj, "Unsupervised automated panorama creation for realistic surveillance scenes through weighted mutual information registration ", SPIE/IS&T: Electronic Imaging Symposium, San Francisco, CA, Jan. 2011.
47. M. S. Erkilinc, M. Jaber, E. Saber and R. Pearson, "Line and Streak Detection on Polished and textured Surfaces using Line Integrals", SPIE/IS&T: Electronic Imaging Symposium, San Francisco, CA, Jan. 2011.
48. M. Jaber, S. Vantaram and E. Saber, "A Probabilistic Framework for Unsupervised Evaluation and Ranking of Image Segmentations", AIPR, Washington, DC, 2010.
49. M. Jaber and E. Saber, "A Bayesian Network-Based Approach for Identifying Regions of Interest Utilizing Global Image Features," Proceedings of the SPIE Optical Engineering + Applications Conference, Vol. 7798, San Diego, CA, Aug. 2010.
50. M. Jaber, E. Saber, M. Shaw, J. Hewitt, "A robust and fast approach for multiple image components stitching," Proceedings of SPIE/IS&T: Electronic Imaging Symposium, San Jose, CA, Jan. 2010.
51. S. R. Vantaram, E. Saber, S. Dianat, M. Shaw and R. Bhaskar, "An Adaptive and Progressive Approach for Efficient Gradient-based Multi-resolution Color Image Segmentation", International Conference on Image Processing, Cairo, Egypt, November 2009.
52. S. Khullar, E. Saber, S. Dianat, J. Trask, R. Lawton, and M. Shaw, "Automatic Multi-resolution Spatio-Frequency Analysis for Print Mottle Evaluation", Proceedings of the SPIE: 17th Color Imaging Conference, Albuquerque, NM, November 2009.
53. P. Gurram, E. Saber, H. Rhody, "Automated 3d Object Identification Using Bayesian Network", AIPR workshop, Washington, D.C., October 2009.
54. X. Fan, H. Rhody and E. Saber, "A Novel Improvement for the HCL-MMI Multi-modal Image Registration by SIFT Algorithm", IEEE Western NY Image Processing Workshop, Sept. 2009.
55. M. Jaber, E. Saber, F. Sahin, "Extraction of Memory Colors Using Bayesian Networks", IEEE SMC International Conference on System of Systems Engineering, Albuquerque, NM, June 2009.
56. S. R. Vantaram, E. Saber, V. Amuso, M. Shaw, R. Bhaskar, "Unsupervised Image Segmentation by Automatic Gradient Thresholding for Dynamic region Growth in the CIE L*a*b* Color Space", SPIE/IS&T: Electronic Imaging Symposium, San Jose, CA, Jan. 2009.
57. X. Fan, H. Rhody and E. Saber, "A Spatial Feature Enhanced MMI Algorithm for Multi-modal Wild Fire Image Registration", AIPR, Washington, DC, 2008.
58. X. Fan, H. Rhody and E. Saber, "A Novel Feature Enhanced MMI based Registration Algorithm for Automated Maps and Images", IEEE International Geoscience & Remote Sensing Symposium, Boston MA, July 2008.
59. L. Garcia, E. Saber, V. Amuso, M. Shaw and R. Bhaskar, "Automatic Color Image Segmentation By Dynamic Region Growth And Multimodal Merging Of Color And Texture Information", International Conference on Acoustics, Speech and Signal Processing, Las Vegas, NV, March 2008.
60. G. Balasubramanian, E. Saber, V. Misic, E. Peskin and M. Shaw, "Unsupervised Color Image Segmentation by Dynamic Color Gradient Thresholding" Proceedings of SPIE/IS&T: Electronic Imaging Symposium, San Jose, CA, Jan. 2008.

61. M. Jaber, E. Saber, S. Dianat, and M. Shaw, “Identification and Ranking of Relevant Image Content”, Proceedings of SPIE/IS&T: Electronic Imaging Symposium, San Jose, CA, Jan. 2008.
62. P. Gurram, E. Saber, H. Rhody, “Extraction of digital elevation map from parallel-perspective stereo mosaics”, Proceedings of SPIE/IS&T: Electronic Imaging Symposium, San Jose, CA, Jan. 2008.
63. X. Fan, H. Rhody and E. Saber, “An Algorithm for Automated Registration of Maps and Images based on Feature Detection and Mutual Information”, Proceedings of SPIE/IS&T: Electronic Imaging Symposium, San Jose, CA, Jan. 2008.
64. P. Gurram, E. Saber, H. Rhody, S. Lach and J. Kerekes, “3D Scene Reconstruction through a Fusion of Passive Video and Lidar Imagery”, AIPR workshop, Washington, D.C., October 2007.
65. K. Chandu, E. Saber and W. Wu, “A Mutual Information Based Automatic Registration And Analysis Algorithm For Defect Identification In Printed Documents”, International Conference on Image Processing, San Antonio, TX, September 2007
66. M. K. Reddy, V. Misic, E. Saber and J. Trask (HP), “A new Adaptive Edge Enhancement Algorithm for Color Laser Printers”, ICASSP, Honolulu, Hawaii, April 2007.
67. X. Fan, H. Rhody and E. Saber, “A Harris Corner Label Enhanced MMI Algorithm for Multi-Modal Airborne Image Registration”, International Conf. on Computer Vision Theory and Appl., Barcelona, March 2007.
68. P. Gurram, E. Saber and H. Rhody, “A Novel Triangulation Method for Building Parallel-Perspective Stereo Mosaics”, Proceedings of SPIE/IS&T: Electronic Imaging Symposium, San Jose, CA, Jan. 2007.
69. M. Husain, E. Saber, V. Misic and S. Joralemon, “Dynamic Object Tracking By Partial Shape Matching For Video Surveillance Applications”, International Conference on Image Proc., Atlanta, GA, October 2006.
70. X. Fan, H. Rhody and E. Saber, “A Comparison of Exhaustive Search vs. Gradient Search for Automatic Imagery Registration based on MMI”, IEEE Western NY Image Processing Workshop, Rochester, NY 2006.
71. O. Ugbeme, E. Saber, and W. Wu, “An Automated Defect Classifying Algorithm for Printed Documents”, International Congress on Imaging Science, Rochester, NY, May 2006.
72. H. Santos, E. Saber, and W. Wu, “A New Algorithm for Streak Detection in Mottle and Noisy Images”, International Congress on Imaging Science, Rochester, NY, May 2006.
73. V. Sampath, V. Misic, E. Saber, H. Liu, and Y. Yu, "Seed Localization Using Trus And Grf Based Gaussian Filtering For Brachytherapy Applications", International Conference on Acoustics, Speech and Signal Processing, Toulouse, France, 2006.
74. X. Fan, H. Rhody and E. Saber, “Automatic Registration of Multi-Sensor Airborne Imagery”, AIPR workshop, Washington, D.C., Oct. 19-21, 2005.
75. K. Modi, F. Sahin and E. Saber, “An Application of Human Robot Interaction: development of a Ping-Pong Playing Robotic Arm”, International Conference on SMC, 2005.
76. A. E. Ononye, A. Vodacek and E. Saber, "Extraction of Active Fire Line and Map Using AVIRIS Imagery", EastFIRE Conference, May 11-13, 2005, Fairfax, VA
77. E. Saber, Y. Xu and A. M. Tekalp, “Object Based Image Labeling using Partial Matching Guided Search”, International Conf. on Acoustics, Speech, and Signal Proc., Philadelphia, USA, 2005.
78. E. Saber, Y. Xu and A. M. Tekalp, “VOOGLE: Tools for Labeling & Manipulating Partial and Full Objects in Images Invariant to Translations, Rotations, Scale and Reflections”, Western New York Workshop, Sept. 2004.
79. Y. Xu, E. Saber, and A. M. Tekalp, “Semantic Object Segmentation by Dynamic Learning from Multiple Examples”, International Conf. on Acoustics, Speech, and Signal Proc., May 2004, Montreal, Canada.
80. M. Celik, G. Sharma, A. M. Tekalp, E. Saber, “Lossless Authentication Watermark”, SPIE/IS&T: EI 2003, San Jose, CA, USA.
81. M. Celik, G. Sharma, A. M. Tekalp, E. Saber, “Reversible Data Hiding”, International Conf. on Image Processing 2002, Rochester, NY, USA.
82. M. Celik, G. Sharma, E. Saber, and A. M. Tekalp, “A Hierarchical Image Authentication Watermark with Improved Localization & Security”, International Conf. on Image Processing 2001, Thessaloniki, Greece.
83. M. Celik, E. Saber, G. Sharma, A. M. Tekalp, “Geometry-Invariant Watermarking”, SPIE/IS&T: Electronic Imaging, San Jose, 2001.
84. Y. Xu, E. Saber, and A. M. Tekalp, “Contour based Shape matching of Partially Occluded Objects for Image Labeling using Hierarchical Content Description”, SPIE:IS&T : Electronic Imaging 2001, San Jose, CA.
85. Y. Xu, E. Saber, and A. M. Tekalp, “Image Retrieval through Shape Matching of Partially Occluded Objects using Hierarchical Content Description”, International Conf. on Image Proc., Vancouver, Canada, 2000.

86. P. Duygulu, Y. Xu, E. Saber, A. M. Tekalp, and F. T. Yarman-Vural, “Object Based Image Retrieval for Multi-Level Segmentation”, International Conf. on Acoustic, Speech, and Signal Processing, Istanbul, Turkey, 2000.
87. Y. Xu, E. Saber, and A. M. Tekalp, “Object-Based Image Retrieval through Learning from User Search Patterns and Profiles”, SPIE/IS&T Electronic Imaging, January 2000, San Jose, California.
88. Y. Xu, E. Saber, and A. M. Tekalp, “Learning-based Hierarchical Content Description for Object Formation and Retrieval”, IEEE Image Processing Workshop, Rochester, NY, September 1999.
89. Y. Xu, E. Saber, and A. M. Tekalp, “Object Formation and Retrieval using a Learning-Based Hierarchical Content Description”, International Conf. on Image Processing, 1999, Kobe, Japan.
90. Y. Xu, E. Saber, and A. M. Tekalp, “Hierarchical Content Description and Object Formation by Learning”, Computer Vision and Pattern Recognition Workshop, Fort Collins, Colorado, 1999.
91. M. Xia, E. Saber, G. Sharma, and A. M. Tekalp, “Total Least Squares Regression in Neugebauer Model Parameter Estimation for Dot-on-Dot Halftone Screens”, Non Impact Printing, October 1998.
92. M. Xia, E. Saber, G. Sharma, and A. M. Tekalp, “Total Least Square Technique in Color Printer Characterization”, International Conference on Image Processing, Chicago, 1998.
93. M. Xia, E. Saber, G. Sharma, and A. M. Tekalp, “Adaptive Content Dependent Color Rendering of Images and Documents”, SPIE/IS&T: Electronic Imaging, San Jose, 1998.
94. M. Xia, E. Saber, G. Sharma, and A. M. Tekalp, “Total least squares regression in color printer calibration”, IEEE Image Processing Workshop, Rochester, NY, September 1997.
95. E. Saber and A. M. Tekalp, “Image Annotation and Retrieval by Integrating Color, Shape and Texture”, International Conference on Image Processing, September 1996, Lausanne, Switzerland.
96. E. Saber and A. M. Tekalp, “Region-Based Image Annotation using Color and Texture Cues”, European Signal Processing Conference, September 1996, Trieste, Italy.
97. E. Saber and A. M. Tekalp, “Detection of Faces and Eyes using Color, Shape and Symmetry-Based Cost Functions”, International Conference on Pattern Recognition, August 1996, Vienna, Austria.
98. E. Saber, A. M. Tekalp, and G. Bozdagi, “Fusion of Color and Edge Information for Improved Segmentation and Edge Linking”, International Conf. on Acoustics, Speech, and Signal Proc., May 1996, Atlanta, Georgia.
99. E. Saber and A. M. Tekalp, “Image Query-by-Example using Region-Based Affine Shape Matching”, SPIE/IS&T Electronic Imaging, Volume 2666, January 1996, San Jose, California.
100. E. Saber, A. M. Tekalp, R. Eschbach and K. Knox, “Annotation Of Natural Scenes Using Adaptive Color Segmentation”, SPIE/IS&T Electronic Imaging, February 1995, San Jose, California.

PATENTS, PATENT APPLICATIONS and PATENT PUBLICATIONS

1. D. Cherry, R. Jessome, R. Maggard, D. Siddall, Y. Raja and E. Saber, “Part Replacement Predictions using Convolutional Neural Networks”, under review.
2. A. Rangnekar, E. Saber, S. Moudgalya, “Interpolating Pixel Values”, Filed February 2016, Patent Pending.
3. P. Lee, E. Saber, O. de Lima, D. Day, P. Bauer, M. Shaw, R. S. Twede, S. Janakiraman, B. Sorensen. “Detecting Document Objects”. # PCT/US2015/053762.
4. S. Vantaram, E. Saber, S. Dianat, M. Shaw and R. Bhaskar, “Methods for Adaptive and Progressive Gradient-Based Multi-resolution Color Image Segmentation and Systems Thereof”, US 8515171B2.
5. M. Shaw, R. Bhaskar, L. Garcia, E. Saber, V. Amuso, “Image Segmentation using Dynamic Color Gradient Threshold, Texture and Multimodal Merging”, US Patent Application Publication US2009/0080773A1.
6. E. Saber, M. Nelson, M. Stauffer, D. Bohan and R. Kothari, “Correcting Artifacts on A Display”, US Patent 9,620,082.
7. M. Shaw, R. Bhaskar, G. Balasubramanian, E. Saber, V. Misic, and E. Peskin “Unsupervised Color Image Segmentation by Dynamic Color Gradient Thresholding”, US Patent 7,873,214.
8. E. Saber & R. Loce, “Corner Sharpening of Text and Line Art in a Super Resolution Anti-Aliasing Image Path”, US Patent 7,536,052.
9. L. K Mestha, S. Bolte, E. Saber, and S. Updegraff, “Systems and Methods for Obtaining a Spatial Color Profile, and Calibrating a Marking System”, US Patent Application Publication 2004/0136013, July 2004.
10. L. K. Mestha, E. Saber, “Systems and methods for sensing marking substrate area coverage using a spectrophotometer”, US Patent 7,110,142 and European EP1309176A2.

11. D. Damji, A. Leon, P. Perez and E. Saber, “Processing system for replaceable modules in a digital printing apparatus”, European Patent EP1079278B.
12. E. Saber, D. Damji, A. Leon, and P. Perez, “Remanufacturing system for replaceable modules in a digital printing apparatus”, US Patent 6173128.

EXTERNAL FUNDING/FUNDED PROPOSALS

Proposals Funded while at RIT

- Acquired/continue to acquire funding with multiple grants from government agencies and corporate partners (Hewlett-Packard, Dataphysics, Lenel, Xerox) in excess of $3.5 Million as PI and in excess of $5 Million as PI/Co-PI (total Portfolio) since joining RIT. A complete statement listing the details of each grant can be provided upon request.

Proposals Funded prior to Joining RIT as an employee of Xerox Corporation and University of Rochester:

- A. M. Tekalp and E. Saber (co-PI), “An intelligent visual database system: Hierarchical content description and matching using integrated similarity metrics”, funded for $243K for four years at the University of Rochester by the National Science Foundation under NSF Grant IIS – 9820721.

HONORS AND AWARDS

- Awarded the Prestigious Trustees Scholarship award – the highest award at RIT with regards to research recognition (2012)
- Elected as the Kate Gleason College of Engineering Scholarship award winner (2012).
- Elected as Electrical and Microelectronic Engineering Gleason Professor for 3 years Fall 2011 – Summer 2014.
- Elected as PI Millionaire by RIT Sponsored research organization.
- Winner of an M.S./Ph.D. scholarship for graduate study from Xerox Corporation.
- Winner of the quality recognition award from Xerox Corporation for outstanding performance.
- Core member of a toner/developer facility engineering project team recognized as ``The 1991 and 1993 team of the year'' by the Delaware Valley Chapter of the Project Management Institute.
- Elected to the Electrical Engineering Honor Society, Eta Kappa Nu.
- Winner of Gibran Khalil Gibran Scholarship for outstanding academic achievements.
- Valedictorian of the Electrical and Computer Engineering Department at the University of Buffalo.
- Recipient of several prizes and awards from the University of Buffalo and the Mohawk Valley Comm. College for excellent academic achievements; and from Xerox Corporation for outstanding performance.

PROFESSIONAL ACTIVITIES

- Senior Member of the Institute of Electrical & Electronic Engineers.
- Member of the IEEE Signal Processing Society.
- Plenary Chair for ICIP 2021.
- Technical Program Chair for ICASSP 2017 in New Orleans.
- General Chair for ICIP 2012.
- Member of the IEEE Signal Processing society conference board for 3 year team starting January 2013.
- Tutorial chair for the International Conferences on Image Processing, ICIP 2007 and ICIP 2009.
- Special session chair on Color Image Processing for the European Signal Processing Conference, Sept. 2006.
- Finance Chair for the International Conference on Image Processing 2002 held in Rochester, NY.
- Former Area Editor for the Journal of Electronic Imaging.
- Guest Editor for the “Color Image Processing” issue of the Signal Processing Magazine.
- Associate Editor for the IEEE Transactions on Image Processing for five years. Term ended: April 2009.

- Former Associate Editor for the IEEE Signal Processing Magazine for DSP Applications Forum.
- Former Member of the IEEE Image & Multidimensional Digital Signal Processing Technical Committee.
- Former Member of IEEE Tech. Comm. on Industry DSP Technology in 2003 & 2004 & Chair for 2005, 2006.
- Technical program committee member for ICIP 2009 and prior ICIPs.
- Technical program committee member for ICASSP 2010 and prior ICASSPs.
- Session chair for several ICIP & ICASSP Conferences and Workshops.
- Chairman, vice-chairman, treasurer, and secretary of the IEEE Rochester Chapter of the signal processing society in 1998, 1997, 1996 and 1995 respectively.
- Reviewer for the IEEE Trans. on Image Processing, IEEE Trans. on Pattern Analysis and Machine Intelligence, Graphical Models and Image Processing, IEEE Trans. on Signal Processing, IEEE Signal Processing Letters, Color Research and Applications, Graphical Modeling and Image Understanding, Image and Vision Computing, Optical Engineering, Journal of Imaging Science and Technology, and the Journal of Electronic Imaging.
- Technical committee for Western New York Imaging Workshop in 1997 and 1999 and general chair in 1998.
- Co-Chair of the Xerox Electronic Image and Video Processing Technology Council.

References Available Upon Request

APPENDIX B

1. U.S. Patent Nos. 10,452,974; 11,238,344; 11,663,474, 10,607,134; 11,113,585; 11,055,583; and 10,102,449.
2. File Histories Corresponding to U.S. Patent Nos. 10,452,974; 11,238,344; 11,663,474, 10,607,134; 11,113,585; 11,055,583; and 10,102,449.
3. https://www.bloomberg.com/news/articles/2021-11-18/apple-accelerates-work-on-car-aims-for-fully-autonomous-vehicle
4. U.S. 2016/0167226
5. Pomerleau, D.A. (1991) Efficient Training of Artificial Neural Networks for Autonomous Navigation. In Neural Computation 3:1 pp. 88-97
6. http://sunnyday.mit.edu/safety-club/workshop5/Adaptive_Cruise_Control_Sys_Overview.pdf
7. https://www.nissan-global.com/EN/INNOVATION/TECHNOLOGY/ARCHIVE/LDW/

Exhibit G

ABSTRACT

Aspects of the disclosure generally relate to computing enabled devices and/or systems, and may be generally directed to devices, systems, methods, and/or applications for learning a device's operation in various circumstances, storing this knowledge in a knowledgebase (i.e. neural network, graph, sequences, etc.), and enabling autonomous operation of the device.

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	
	Application Number	
Title of Invention	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION	

The application data sheet is part of the provisional or nonprovisional application for which it is being submitted. The following form contains the bibliographic data arranged in a format specified by the United States Patent and Trademark Office as outlined in 37 CFR 1.76.
This document may be completed electronically and submitted to the Office in electronic format using the Electronic Filing System (EFS) or the document may be printed and included in a paper filed application.

Secrecy Order 37 CFR 5.2:

☐ Portions or all of the application associated with this Application Data Sheet may fall under a Secrecy Order pursuant to 37 CFR 5.2 (Paper filers only. Applications that fall under Secrecy Order may not be filed electronically.)

Inventor Information:

Inventor 1 Remove

Legal Name

Prefix	**Given Name**	**Middle Name**	**Family Name**	**Suffix**
	Jasmin		Cosic	

Residence Information (Select One) ● US Residency ○ Non US Residency ○ Active US Military Service

City	Miami	**State/Province**	FL	**Country of Residence**	US

Mailing Address of Inventor:

Address 1	108 Woodbury Street		
Address 2			
City	Pawtucket	**State/Province**	RI
Postal Code	02861	**Country**	US

All Inventors Must Be Listed - Additional Inventor Information blocks may be generated within this form by selecting the **Add** button. Add

Correspondence Information:

Enter either Customer Number or complete the Correspondence Information section below. For further information see 37 CFR 1.33(a).

☐ **An Address is being provided for the correspondence Information of this application.**

Customer Number	116094	
Email Address	cpapc29@hotmail.com	Add Email Remove Email

Application Information:

Title of the Invention	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION		
Attorney Docket Number		**Small Entity Status Claimed**	☒
Application Type	Nonprovisional		
Subject Matter	Utility		
Total Number of Drawing Sheets (if any)	40	**Suggested Figure for Publication (if any)**	2

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	
	Application Number	
Title of Invention	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION	

Filing By Reference:

Only complete this section when filing an application by reference under 35 U.S.C. 111(c) and 37 CFR 1.57(a). Do not complete this section if application papers including a specification and any drawings are being filed. Any domestic benefit or foreign priority information must be provided in the appropriate section(s) below (i.e., "Domestic Benefit/National Stage Information" and "Foreign Priority Information").

For the purposes of a filing date under 37 CFR 1.53(b), the description and any drawings of the present application are replaced by this reference to the previously filed application, subject to conditions and requirements of 37 CFR 1.57(a).

Application number of the previously filed application	Filing date (YYYY-MM-DD)	Intellectual Property Authority or Country

Publication Information:

☐ Request Early Publication (Fee required at time of Request 37 CFR 1.219)

☒ **Request Not to Publish.** I hereby request that the attached application not be published under 35 U.S.C. 122(b) and certify that the invention disclosed in the attached application **has not and will not** be the subject of an application filed in another country, or under a multilateral international agreement, that requires publication at eighteen months after filing.

Representative Information:

Representative information should be provided for all practitioners having a power of attorney in the application. Providing this information in the Application Data Sheet does not constitute a power of attorney in the application (see 37 CFR 1.32).
Either enter Customer Number or complete the Representative Name section below. If both sections are completed the customer Number will be used for the Representative Information during processing.

Please Select One:	● Customer Number	US Patent Practitioner	○ Limited Recognition (37 CFR 11.9)
Customer Number			

Domestic Benefit/National Stage Information:

This section allows for the applicant to either claim benefit under 35 U.S.C. 119(e), 120, 121, 365(c), or 386(c) or indicate National Stage entry from a PCT application. Providing benefit claim information in the Application Data Sheet constitutes the specific reference required by 35 U.S.C. 119(e) or 120, and 37 CFR 1.78.
When referring to the current application, please leave the "Application Number" field blank.

Prior Application Status			Remove
Application Number	Continuity Type	Prior Application Number	Filing or 371(c) Date (YYYY-MM-DD)

Additional Domestic Benefit/National Stage Data may be generated within this form by selecting the **Add** button. Add

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	
	Application Number	
Title of Invention	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION	

Foreign Priority Information:

This section allows for the applicant to claim priority to a foreign application. Providing this information in the application data sheet constitutes the claim for priority as required by 35 U.S.C. 119(b) and 37 CFR 1.55. When priority is claimed to a foreign application that is eligible for retrieval under the priority document exchange program (PDX)[i] the information will be used by the Office to automatically attempt retrieval pursuant to 37 CFR 1.55(i)(1) and (2). Under the PDX program, applicant bears the ultimate responsibility for ensuring that a copy of the foreign application is received by the Office from the participating foreign intellectual property office, or a certified copy of the foreign priority application is filed, within the time period specified in 37 CFR 1.55(g)(1).

Remove

Application Number	Country[i]	Filing Date (YYYY-MM-DD)	Access Code[i] (if applicable)

Additional Foreign Priority Data may be generated within this form by selecting the **Add** button.

Add

Statement under 37 CFR 1.55 or 1.78 for AIA (First Inventor to File) Transition Applications

☐ This application (1) claims priority to or the benefit of an application filed before March 16, 2013 and (2) also contains, or contained at any time, a claim to a claimed invention that has an effective filing date on or after March 16, 2013.
NOTE: By providing this statement under 37 CFR 1.55 or 1.78, this application, with a filing date on or after March 16, 2013, will be examined under the first inventor to file provisions of the AIA.

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	
	Application Number	

Title of Invention	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION

Authorization or Opt-Out of Authorization to Permit Access:

When this Application Data Sheet is properly signed and filed with the application, applicant has provided written authority to permit a participating foreign intellectual property (IP) office access to the instant application-as-filed (see paragraph A in subsection 1 below) and the European Patent Office (EPO) access to any search results from the instant application (see paragraph B in subsection 1 below).

Should applicant choose not to provide an authorization identified in subsection 1 below, applicant **must opt-out** of the authorization by checking the corresponding box A or B or both in subsection 2 below.

NOTE: This section of the Application Data Sheet is **ONLY** reviewed and processed with the **INITIAL** filing of an application. After the initial filing of an application, an Application Data Sheet cannot be used to provide or rescind authorization for access by a foreign IP office(s). Instead, Form PTO/SB/39 or PTO/SB/69 must be used as appropriate.

1. Authorization to Permit Access by a Foreign Intellectual Property Office(s)

A. Priority Document Exchange (PDX) - Unless box A in subsection 2 (opt-out of authorization) is checked, the undersigned hereby **grants the USPTO authority** to provide the European Patent Office (EPO), the Japan Patent Office (JPO), the Korean Intellectual Property Office (KIPO), the State Intellectual Property Office of the People's Republic of China (SIPO), the World Intellectual Property Organization (WIPO), and any other foreign intellectual property office participating with the USPTO in a bilateral or multilateral priority document exchange agreement in which a foreign application claiming priority to the instant patent application is filed, access to: (1) the instant patent application-as-filed and its related bibliographic data, (2) any foreign or domestic application to which priority or benefit is claimed by the instant application and its related bibliographic data, and (3) the date of filing of this Authorization. See 37 CFR 1.14(h)(1).

B. Search Results from U.S. Application to EPO - Unless box B in subsection 2 (opt-out of authorization) is checked, the undersigned hereby **grants the USPTO authority** to provide the EPO access to the bibliographic data and search results from the instant patent application when a European patent application claiming priority to the instant patent application is filed. See 37 CFR 1.14(h)(2).

The applicant is reminded that the EPO's Rule 141(1) EPC (European Patent Convention) requires applicants to submit a copy of search results from the instant application without delay in a European patent application that claims priority to the instant application.

2. Opt-Out of Authorizations to Permit Access by a Foreign Intellectual Property Office(s)

☒ A. Applicant **DOES NOT** authorize the USPTO to permit a participating foreign IP office access to the instant application-as-filed. If this box is checked, the USPTO will not be providing a participating foreign IP office with any documents and information identified in subsection 1A above.

☒ B. Applicant **DOES NOT** authorize the USPTO to transmit to the EPO any search results from the instant patent application. If this box is checked, the USPTO will not be providing the EPO with search results from the instant application.

NOTE: Once the application has published or is otherwise publicly available, the USPTO may provide access to the application in accordance with 37 CFR 1.14.

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	
	Application Number	
Title of Invention	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION	

Applicant Information:

Providing assignment information in this section does not substitute for compliance with any requirement of part 3 of Title 37 of CFR to have an assignment recorded by the Office.

Applicant 1 Remove

If the applicant is the inventor (or the remaining joint inventor or inventors under 37 CFR 1.45), this section should not be completed. The information to be provided in this section is the name and address of the legal representative who is the applicant under 37 CFR 1.43; or the name and address of the assignee, person to whom the inventor is under an obligation to assign the invention, or person who otherwise shows sufficient proprietary interest in the matter who is the applicant under 37 CFR 1.46. If the applicant is an applicant under 37 CFR 1.46 (assignee, person to whom the inventor is obligated to assign, or person who otherwise shows sufficient proprietary interest) together with one or more joint inventors, then the joint inventor or inventors who are also the applicant should be identified in this section.

Clear

Assignee	Legal Representative under 35 U.S.C. 117	Joint Inventor
Person to whom the inventor is obligated to assign.	Person who shows sufficient proprietary interest	

If applicant is the legal representative, indicate the authority to file the patent application, the inventor is:

Name of the Deceased or Legally Incapacitated Inventor:

If the Applicant is an Organization check here. ☐

Prefix	**Given Name**	Middle Name	**Family Name**	Suffix

Mailing Address Information For Applicant:

Address 1			
Address 2			
City		**State/Province**	
Country		Postal Code	
Phone Number		Fax Number	
Email Address			

Additional Applicant Data may be generated within this form by selecting the Add button. Add

Assignee Information including Non-Applicant Assignee Information:

Providing assignment information in this section does not substitute for compliance with any requirement of part 3 of Title 37 of CFR to have an assignment recorded by the Office.

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	
	Application Number	

Title of Invention	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION

Assignee 1

Complete this section if assignee information, including non-applicant assignee information, is desired to be included on the patent application publication. An assignee-applicant identified in the "Applicant Information" section will appear on the patent application publication as an applicant. For an assignee-applicant, complete this section only if identification as an assignee is also desired on the patent application publication.

Remove

If the Assignee or Non-Applicant Assignee is an Organization check here. ☐

Prefix	**Given Name**	Middle Name	**Family Name**	Suffix

Mailing Address Information For Assignee including Non-Applicant Assignee:

Address 1			
Address 2			
City		**State/Province**	
Country i		Postal Code	
Phone Number		Fax Number	
Email Address			

Additional Assignee or Non-Applicant Assignee Data may be generated within this form by selecting the Add button. Add

Signature:

Remove

NOTE: This Application Data Sheet must be signed in accordance with 37 CFR 1.33(b). **However, if this Application Data Sheet is submitted with the INITIAL filing of the application and either box A or B is not checked in subsection 2 of the "Authorization or Opt-Out of Authorization to Permit Access" section, then this form must also be signed in accordance with 37 CFR 1.14(c).**

This Application Data Sheet **must** be signed by a patent practitioner if one or more of the applicants is a **juristic entity** (e.g., corporation or association). If the applicant is two or more joint inventors, this form must be signed by a patent practitioner, **all** joint inventors who are the applicant, or one or more joint inventor-applicants who have been given power of attorney (e.g., see USPTO Form PTO/AIA/81) on behalf of **all** joint inventor-applicants.

See 37 CFR 1.4(d) for the manner of making signatures and certifications.

Signature	/Jasmin Cosic/		Date (YYYY-MM-DD)	2016-11-02
First Name	Jasmin	Last Name	Cosic	Registration Number	

Additional Signature may be generated within this form by selecting the Add button. Add

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	
	Application Number	

Title of Invention	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1 The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these records.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3 A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent CooperationTreaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

PTO/AIA/01 (06-12)
Approved for use through 01/31/2014. OMB 0651-0032
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

DECLARATION (37 CFR 1.63) FOR UTILITY OR DESIGN APPLICATION USING AN APPLICATION DATA SHEET (37 CFR 1.76)

Title of Invention	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION

As the below named inventor, I hereby declare that:

This declaration is directed to:

- [x] The attached application, or
- [ ] United States application or PCT international application number ____________________

  filed on ____________________.

The above-identified application was made or authorized to be made by me.

I believe that I am the original inventor or an original joint inventor of a claimed invention in the application.

I hereby acknowledge that any willful false statement made in this declaration is punishable under 18 U.S.C. 1001 by fine or imprisonment of not more than five (5) years, or both.

WARNING:

Petitioner/applicant is cautioned to avoid submitting personal information in documents filed in a patent application that may contribute to identity theft. Personal information such as social security numbers, bank account numbers, or credit card numbers (other than a check or credit card authorization form PTO-2038 submitted for payment purposes) is never required by the USPTO to support a petition or an application. If this type of personal information is included in documents submitted to the USPTO, petitioners/applicants should consider redacting such personal information from the documents before submitting them to the USPTO. Petitioner/applicant is advised that the record of a patent application is available to the public after publication of the application (unless a non-publication request in compliance with 37 CFR 1.213(a) is made in the application) or issuance of a patent. Furthermore, the record from an abandoned application may also be available to the public if the application is referenced in a published application or an issued patent (see 37 CFR 1.14). Checks and credit card authorization forms PTO-2038 submitted for payment purposes are not retained in the application file and therefore are not publicly available.

LEGAL NAME OF INVENTOR

Inventor: Jasmin Cosic   Date (Optional) : ____________________

Signature: ____________________

Note: An application data sheet (PTO/SB/14 or equivalent), including naming the entire inventive entity, must accompany this form or must have been previously filed. Use an additional PTO/AIA/01 form for each additional inventor.

ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION

FIELD

The disclosure generally relates to computing enabled devices and/or systems. The disclosure includes devices, apparatuses, systems, and related methods for providing advanced learning, anticipating, decision making, automation, and/or other functionalities.

COPYRIGHT NOTICE

BACKGROUND

Devices or systems commonly operate by receiving a user's operating directions in various circumstances. Instructions are then executed to effect the operation of a device or system based on user's operating directions. Hence, devices or systems rely on the user to direct their behaviors. Commonly employed device or system operating techniques lack a way to learn operation of a device or system and enable autonomous operation of a device or system.

SUMMARY

In some aspects, the disclosure relates to a system for learning and using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises a processor circuit configured to execute instruction sets for operating a device. The system may further include a memory unit configured to store data. The system may further include a sensor configured to detect objects. The system may further include an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first collection of object representations, the first collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to learn the first collection of object representations correlated with the first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: receive a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit,

wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations performed in response to the executing by the processor circuit.

In some embodiments, at least one of: the processor circuit, the memory unit, the sensor, or the artificial intelligence unit are part of, operating on, or coupled to the device. In further embodiments, the device includes one or more devices. In further embodiments, the device includes a construction machine, an assembly machine, an object handling machine, an object dispensing machine, a sorting machine, a restocking machine, an industrial machine, an agricultural machine, a harvesting machine, an appliance, a toy, a robot, a ground vehicle, an aerial vehicle, an aquatic vehicle, a computer, a smartphone, a control device, or a computing enabled device. In further embodiments, the processor circuit includes one or more processor circuits. In further embodiments, the processor circuit includes a logic circuit. The logic circuit may include a microcontroller. The one or more instruction sets may include one or more inputs into or one or more outputs from the logic circuit.

In certain embodiments, the processor circuit includes a logic circuit, the instruction sets for operating the device include inputs into the logic circuit, and executing instruction sets for operating the device includes performing logic operations on the inputs into the logic circuit and producing outputs for operating the device. The logic circuit includes a microcontroller.

In some embodiments, the processor circuit includes a logic circuit, the instruction sets for operating the device include outputs from the logic circuit for operating the device, and executing instruction sets for operating the device includes performing logic operations on inputs into the logic circuit and producing the outputs from the logic circuit for operating the device.

In certain embodiments, the memory unit includes one or more memory units. In further embodiments, the memory unit resides on a remote computing device or a remote computing system, the remote computing device or the remote computing system coupled to the processor circuit via a network or an interface. The remote computing device or the remote computing system may include a server, a cloud, a computing device, or a computing system accessible over the network or the interface.

In some embodiments, the sensor includes one or more sensors. In further embodiments, the sensor includes a camera, a microphone, a lidar, a radar, a sonar, or a detector. In further embodiments, the sensor is part of a remote device. In further embodiments, the sensor is configured to detect objects in the device's surrounding.

In certain embodiments, the artificial intelligence unit is coupled to the sensor. In further embodiments, the artificial intelligence unit is coupled to the memory unit. In further embodiments, the artificial intelligence unit is part of, operating on, or coupled to the processor circuit. In further embodiments, the system further comprises: an additional processor circuit, wherein the artificial intelligence unit is part of, operating on, or coupled to the additional processor circuit. In further embodiments, the artificial intelligence unit is part of, operating on, or coupled to a remote computing device or a remote computing system, the remote computing device or the remote computing system coupled to the processor circuit via a network or an interface. In further embodiments, the artificial intelligence unit includes a circuit, a computing apparatus, or a computing system attachable to the processor circuit. In further embodiments, the artificial intelligence unit includes a circuit, a computing apparatus, or a computing system attachable to the device. In further embodiments, the artificial intelligence unit is attachable to an application for operating the device, the application running on the processor circuit. In further embodiments, the artificial

intelligence unit includes a circuit, a computing apparatus, or a computing system built into the processor circuit. In further embodiments, the artificial intelligence unit includes a circuit, a computing apparatus, or a computing system built into the device. In further embodiments, the artificial intelligence unit is built into an application for operating the device, the application running on the processor circuit. In further embodiments, the artificial intelligence unit is provided as a feature of the processor circuit. In further embodiments, the artificial intelligence unit is provided as a feature of an application running on the processor circuit. In further embodiments, the artificial intelligence unit is provided as a feature of the device. In further embodiments, the artificial intelligence unit is further configured to: take control from, share control with, or release control to the processor circuit. In further embodiments, the artificial intelligence unit is further configured to: take control from, share control with, or release control to an application or an object of the application, the application running on the processor circuit.

In some embodiments, the first collection of object representations includes one or more representations of objects detected by the sensor at a time. In further embodiments, the new collection of object representations includes one or more representations of objects detected by the sensor at a time. In further embodiments, the first collection of object representations includes a stream of collections of object representations. In further embodiments, the new collection of object representations includes a stream of collections of object representations. In further embodiments, the first or the new collection of object representations includes or is associated with a time stamp, an order, or a time related information. In further embodiments, the first or the new collection of object representations includes one or more representations of objects in the device's surrounding. In further embodiments, the first or the new collection of object representations includes one or more representations of objects in a remote device's surrounding. In further embodiments, an object representation of the one or more object representations includes one or more object properties. In further embodiments, the first or the new collection of object representations includes one or more object properties. In further embodiments, the first collection of object representations includes a comparative collection of object representations whose at least one portion can be used for comparisons with at least one portion of collections of object representations subsequent to the first collection of object representations, the collections of object representations subsequent to the first collection of object representations comprising the new collection of object representations. In further embodiments, the first collection of object representations includes a comparative collection of object representations that can be used for comparisons with the new collection of object representations. In further embodiments, the new collection of object representations includes an anticipatory collection of object representations whose correlated one or more instruction sets can be used for anticipation of one or more instruction sets to be executed by the processor circuit.

In certain embodiments, the first one or more instruction sets for operating the device include one or more instruction sets that temporally correspond to the first collection of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first collection of object representations include one or more instruction sets executed at a time of generating the first collection of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first collection of object representations include one or more instruction sets executed prior to generating the first collection of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first collection of object representations include one or more instruction sets executed within a threshold period of time prior to generating the first collection of object representations. In further embodiments, the one or more instruction

sets that temporally correspond to the first collection of object representations include one or more instruction sets executed subsequent to generating the first collection of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first collection of object representations include one or more instruction sets executed within a threshold period of time subsequent to generating the first collection of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first collection of object representations include one or more instruction sets executed within a threshold period of time prior to generating the first collection of object representations or a threshold period of time subsequent to generating the first collection of object representations.

In some embodiments, the first one or more instruction sets for operating the device are executed by the processor circuit. In further embodiments, the first one or more instruction sets for operating the device are part of an application for operating the device, the application running on the processor circuit. In further embodiments, the first one or more instruction sets for operating the device include one or more inputs into or one or more outputs from the processor circuit. In further embodiments, the first one or more instruction sets for operating the device include values or states of one or more registers or elements of the processor circuit. In further embodiments, the first one or more instruction sets for operating the device include at least one of: a command, a keyword, a symbol, an instruction, an operator, a variable, a value, an object, a data structure, a function, a parameter, a state, a signal, an input, an output, a character, a digit, or a reference thereto. In further embodiments, the first one or more instruction sets for operating the device include a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, a structured query language (SQL) code, or a machine code. In further embodiments, the first one or more instruction sets for operating the device include one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the processor circuit includes a logic circuit. The first one or more instruction sets for operating the device may include one or more inputs into a logic circuit. The first one or more instruction sets for operating the device may include one or more outputs from a logic circuit.

In certain embodiments, the first one or more instruction sets for operating the device include one or more instruction sets for operating an application or an object of the application, the application running on the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes obtaining the first one or more instruction sets for operating the device from the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device as they are executed by the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from a register or an element of the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from an element that is part of, operating on, or coupled to the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from at least one of: the memory unit, the device, a virtual machine, a runtime engine, a hard drive, a storage device, a peripheral device, a network connected device, or a user. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from a plurality of

processor circuits, applications, memory units, devices, virtual machines, runtime engines, hard drives, storage devices, peripheral devices, network connected devices, or users.

In some embodiments, the processor circuit includes a logic circuit, and wherein the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from the logic circuit. In further embodiments, the logic circuit includes a microcontroller. In further embodiments, the receiving the first one or more instruction sets for operating the device from the logic circuit includes receiving the first one or more instruction sets for operating the device from an element of the logic circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device from the logic circuit includes receiving one or more inputs into the logic circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device from the logic circuit includes receiving one or more outputs from the logic circuit.

In certain embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from an application for operating the device, the application running on the processor circuit.

In some embodiments, the system further comprises: an application including instruction sets for operating the device, the application running on the processor circuit, wherein the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from the application.

In certain embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, a structured query language (SQL) code, or a machine code. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of an element that is part of, operating on, or coupled to the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a register of the processor circuit, the memory unit, a storage, or a repository where the first one or more instruction sets for operating the device are stored. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of the processor circuit, the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a processing element. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of the processor circuit or tracing, profiling, or instrumentation of a component of the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of an application or an object of the application, the application running on the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the

receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of one or more of code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a user input. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: a manual, an automatic, a dynamic, or a just in time (JIT) tracing, profiling, or instrumentation. In further embodiments, the receiving the first one or more instruction sets for operating the device includes utilizing at least one of: a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, a logging tool, or an independent tool for obtaining instruction sets. In further embodiments, the receiving the first one or more instruction sets for operating the device includes utilizing an assembly language. In further embodiments, the receiving the first one or more instruction sets for operating the device includes utilizing a branch or a jump. In further embodiments, the receiving the first one or more instruction sets for operating the device includes a branch tracing or a simulation tracing.

In further embodiments, the system further comprises: an interface configured to receive instruction sets, wherein the first one or more instruction sets for operating the device are received by the interface. The interface may include an acquisition interface.

In some embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the device include a knowledge of how the device operated in a circumstance. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the device are included in a neuron, a node, a vertex, or an element of a knowledgebase. In further embodiments, the knowledgebase includes a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. In further embodiments, some of the neurons, nodes, vertices, or elements are interconnected. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the device are structured into a knowledge cell. In further embodiments, the knowledge cell is included in a neuron, a node, a vertex, or an element of a knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device includes correlating the first collection of object representations with the first one or more instruction sets for operating the device. In further embodiments, the correlating the first collection of object representations with the first one or more instruction sets for operating the device includes generating a knowledge cell, the knowledge cell comprising the first collection of object representations correlated with the first one or more instruction sets for operating the device. In further embodiments, the correlating the first collection of object representations with the first one or more instruction sets for operating the device includes structuring a knowledge of how the device operated in a circumstance. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device includes learning a user's knowledge, style, or methodology of operating the device in a circumstance.

In certain embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device includes storing the first collection of object representations

correlated with the first one or more instruction sets for operating the device into the memory unit, the first collection of object representations correlated with the first one or more instruction sets for operating the device being part of a plurality of collections of object representations correlated with one or more instruction sets for operating the device stored in the memory unit. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the device include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the device are organized into a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, each of the plurality of collections of object representations correlated with one or more instruction sets for operating the device is included in a neuron, a node, a vertex, or an element of a knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the device include a user's knowledge, style, or methodology of operating the device in circumstances. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the device are stored on a remote computing device or a remote computing system. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the device include an artificial intelligence system for knowledge structuring, storing, or representation. The artificial intelligence system for knowledge structuring, storing, or representation may include at least one of: a deep learning system, a supervised learning system, an unsupervised learning system, a neural network, a search-based system, an optimization-based system, a logic-based system, a fuzzy logic-based system, a tree-based system, a graph-based system, a hierarchical system, a symbolic system, a sub-symbolic system, an evolutionary system, a genetic system, a multi-agent system, a deterministic system, a probabilistic system, or a statistical system.

In some embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes comparing at least one portion of the new collection of object representations with at least one portion of the first collection of object representations. In further embodiments, the at least one portion of the new collection of object representations include at least one object representation or at least one object property of the new collection of object representations. In further embodiments, the at least one portion of the first collection of object representations include at least one object representation or at least one object property of the first collection of object representations. In further embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes comparing at least one object representation from the new collection of object representations with at least one object representation from the first collection of object representations. In further embodiments, the comparing at least one object representation from the new collection of object representations with at least one object representation from the first

collection of object representations includes comparing at least one object property of the at least one object representation from the new collection of object representations with at least one object property of the at least one object representation from the first collection of object representations.

In certain embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes determining that there is at least a partial match between the new collection of object representations and the first collection of object representations. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that there is at least a partial match between one or more portions of the new collection of object representations and one or more portions of the first collection of object representations. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that a similarity between at least one portion of the new collection of object representations and at least one portion of the first collection of object representations exceeds a similarity threshold. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining a substantial similarity between at least one portion of the new collection of object representations and at least one portion of the first collection of object representations. The substantial similarity may be achieved when a similarity between the at least one portion of the new collection of object representations and the at least one portion of the first collection of object representations exceeds a similarity threshold. The substantial similarity may be achieved when a number or a percentage of matching or partially matching portions of the new collection of object representations and portions of the first collection of object representations exceeds a threshold number or threshold percentage. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that a number or a percentage of matching or partially matching object representations from the new collection of object representations and from the first collection of object representations exceeds a threshold number or threshold percentage. The matching or partially matching object representations from the new collection of object representations and from the first collection of object representations may be determined factoring in at least one of: a type of an object representation, an importance of an object representation, a threshold for a similarity in an object representation, or a threshold for a difference in an object representation. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that a number or a percentage of matching or partially matching object properties from the new collection of object representations and from the first collection of object representations exceeds a threshold number or threshold percentage. The matching or partially matching object properties from the new collection of object representations and from the first collection of object representations may be determined factoring in at least one of: an association of an object property with an object representation, a category of an object property, an importance of an object property, a threshold for a similarity in an object property, or a threshold for a difference in an object property. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that there

is at least a partial match between at least one object representation from the new collection of object representations and at least one object representation from the first collection of object representations. The determining that there is at least a partial match between at least one object representation from the new collection of object representations and at least one object representation from the first collection of object representations includes determining that there is at least a partial match between at least one object property of the at least one object representation from the new collection of object representations and at least one object property of the at least one object representation from the first collection of object representations.

In certain embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations instead of or prior to an instruction set that would have been executed next. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first collection of object representations into a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting the processor circuit to the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting the processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes transmitting, to the processor circuit for execution, the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes issuing an interrupt to the processor circuit and executing the first one or more instruction sets for operating the device correlated with the first collection of object representations following the interrupt. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying an element that is part of, operating on, or coupled to the processor circuit.

In some embodiments, the processor circuit includes a logic circuit, and wherein the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the logic circuit

includes a microcontroller. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying an element of the logic circuit. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first collection of object representations into an element of the logic circuit. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting the logic circuit to the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes replacing inputs into the logic circuit with the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes replacing outputs from the logic circuit with the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In certain embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes causing an application for operating the device to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations, the application running on the processor circuit.

In some embodiments, the system further comprises: an application including instruction sets for operating the device, the application running on the processor circuit, wherein the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying the application.

In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting an application to the first one or more instruction sets for operating the device correlated with the first collection of object representations, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting an application to one or more alternate instruction sets, the application running on the processor circuit, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying at least one of: the memory unit, a register of the processor circuit, a

storage, or a repository where instruction sets are stored or used. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets for operating an application or an object of the application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying at least one of: an element of the processor circuit, an element of the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing an assembly language. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes adding or inserting additional code into a code of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or

more instruction sets for operating the device correlated with the first collection of object representations includes at least one of: modifying, removing, rewriting, or overwriting a code of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of an application, the application running on the processor circuit. The branching or redirecting the code may include inserting at least one of: a branch, a jump, or a means for redirecting an execution. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes implementing a user's knowledge, style, or methodology of operating the device in a circumstance.

In certain embodiments, the system further comprises: an interface configured to cause execution of instruction sets, wherein the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations is caused by the interface. The interface may include a modification interface.

In some embodiments, the one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations include one or more operations with or by a computing enabled device. In further embodiments, the performing the one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations includes implementing a user's knowledge, style, or methodology of operating the device in a circumstance.

In certain embodiments, the system further comprising: an application running on the processor circuit.

In some embodiments, the instruction sets for operating the device are part of an application for operating the device, the application running on the processor circuit.

In certain embodiments, the system of further comprises: an application for operating the device, the application running on the processor circuit. The application for operating the device may include the instruction sets for operating the device.

In some embodiments, the artificial intelligence unit is further configured to: receive at least one extra information. In further embodiments, the at least one extra information include one or more of: a time information, a location information, a computed information, or a contextual information. In further embodiments, the at least one extra information include one or more of: an information on an object, an information on an object representation, an information on a collection of object representations, an information on a device's circumstance, an information on an instruction set, an information on an application, an information on the processor circuit, an information on the device, or an information on an user. In further embodiments, the artificial intelligence unit is further configured to: learn the first collection of object representations correlated with the at least one extra information. The learning the first collection of object representations correlated with at least one extra information may include correlating the first collection of object representations with the at least one extra information. The learning the first collection of object representations correlated with at least one extra information may include storing the first collection of object representations correlated with the at least one extra information into the memory unit. In further embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first

collection of object representations includes anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations. The anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations may include comparing an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations. The anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations may include determining that a similarity between an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations exceeds a similarity threshold.

In certain embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: cause the user interface to present a user with an option to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In some embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: receive, via the user interface, a user's selection to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In certain embodiments, the artificial intelligence unit is further configured to: rate the executed first one or more instruction sets for operating the device correlated with the first collection of object representations. The rating the executed first one or more instruction sets for operating the device correlated with the first collection of object representations may include causing a user interface to display the executed first one or more instruction sets for operating the device correlated with the first collection of object representations along with one or more rating values as options to be selected by a user. The rating the executed first one or more instruction sets for operating the device correlated with the first collection of object representations may include rating the executed first one or more instruction sets for operating the device correlated with the first collection of object representations without a user input.

In some embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: cause the user interface to present a user with an option to cancel the execution of the executed first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the canceling the execution of the executed first one or more instruction sets for operating the device correlated with the first collection of object representations includes restoring the processor circuit or the device to a prior state. The restoring the processor circuit or the device to a prior state may include saving the state of the processor circuit or the device prior to executing the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In certain embodiments, the system further comprises: an input device configured to receive a user's operating directions, the user's operating directions for instructing the processor circuit on how to operate the device.

In some embodiments, the autonomous device operating includes a partially or a fully autonomous device operating. The partially autonomous device operating may include executing the first one or more instruction sets for operating the device correlated with the first collection of object representations responsive to a user confirmation. In further embodiments, the fully autonomous device operating may include executing the first one or more instruction sets for operating the device correlated with the first collection of object representations without a user confirmation.

In certain embodiments, the artificial intelligence unit is further configured to: receive a second collection of object representations, the second collection of object representations including one or more representations of objects detected by the sensor; receive a second one or more instruction sets for operating the device; and learn the second collection of object representations correlated with the second one or more instruction sets for operating the device. In further embodiments, the second collection of object representations includes one or more representations of objects detected by the sensor at a time. In further embodiments, the second collection of object representations includes a stream of collections of object representations. In further embodiments, the second collection of object representations includes or is associated with a time stamp, an order, or a time related information. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device and the learning the second collection of object representations correlated with the second one or more instruction sets for operating the device include creating a connection between the first collection of object representations correlated with the first one or more instruction sets for operating the device and the second collection of object representations correlated with the second one or more instruction sets for operating the device. The connection may include or be associated with at least one of: an occurrence count, a weight, a parameter, or a data. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device and the learning the second collection of object representations correlated with the second one or more instruction sets for operating the device include updating a connection between the first collection of object representations correlated with the first one or more instruction sets for operating the device and the second collection of object representations correlated with the second one or more instruction sets for operating the device. The updating the connection between the first collection of object representations correlated with the first one or more instruction sets for operating the device and the second collection of object representations correlated with the second one or more instruction sets for operating the device may include updating at least one of: an occurrence count, a weight, a parameter, or a data included in or associated with the connection. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device includes storing the first collection of object representations correlated with the first one or more instruction sets for operating the device into a first node of a knowledgebase, and wherein the learning the second collection of object representations correlated with the second one or more instruction sets for operating the device includes storing the second collection of object representations correlated with the second one or more instruction sets for operating the device into a second node of the knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. The knowledgebase may be stored in the memory unit. The learning the first collection of object representations correlated with the first one or more instruction sets for operating the device and the learning the second collection of object representations correlated with the second one or more instruction sets for operating the device include creating a connection

between the first node and the second node. The learning the first collection of object representations correlated with the first one or more instruction sets for operating the device and the learning the second collection of object representations correlated with the second one or more instruction sets for operating the device include updating a connection between the first node and the second node. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the device is stored into a first node of a neural network and the second collection of object representations correlated with the second one or more instruction sets for operating the device is stored into a second node of the neural network. The first node and the second node may be connected by a connection. The first node may be part of a first layer of the neural network and the second node may be part of a second layer of the neural network. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the device is stored into a first node of a graph and the second collection of object representations correlated with the second one or more instruction sets for operating the device is stored into a second node of the graph. The first node and the second node may be connected by a connection. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the device is stored into a first node of a sequence and the second collection of object representations correlated with the second one or more instruction sets for operating the device is stored into a second node of the sequence.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first collection of object representations, the first collection of object representations including one or more representations of objects detected by a sensor. The operations may further include: receiving a first one or more instruction sets for operating a device. The operations may further include: learning the first collection of object representations correlated with the first one or more instruction sets for operating the device. The operations may further include: receiving a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The operations may further include: anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further include: causing an execution of the first one or more instruction sets for operating the device correlated with the first collection of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations by the device performed in response to the execution.

In certain embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from the one or more processor circuits or from another one or more processor circuits. In further embodiments, the execution of the first one or more instruction sets for operating the device correlated with the first collection of object representations is performed by the one or more processor circuits or by another one or more processor circuits.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a processor circuit, the first collection of object representations including one or more representations of objects detected by a sensor. The method may further include: (b) receiving a first one or more instruction sets for operating a device by the processor circuit. The method may further include: (c) learning the first collection of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the processor circuit. The method may further include: (d) receiving a new collection of object representations by the processor circuit, the new collection of object representations including one or more representations of objects detected by the sensor. The method may further include: (e) anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (e) performed by the processor circuit. The method may further include: (f) executing the first one or more instruction sets for operating the device correlated with the first collection of object representations, the executing of (f) performed in response to the anticipating of (e). The method may further include: (g) performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations by the device performed in response to the executing of (f).

In certain embodiments, the receiving of (b) includes receiving the first one or more instruction sets for operating the device from the processor circuit or from another processor circuit. In further embodiments, the executing of (f) is performed by the processor circuit or by another processor circuit.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable as well as the following embodiments.

In some embodiments, the first one or more instruction sets for operating the device are executed by a processor circuit. In further embodiments, the first one or more instruction sets for operating the device are part of an application for operating the device. In further embodiments, the first one or more instruction sets for operating the device include one or more inputs into or one or more outputs from a processor circuit. In further embodiments, the first one or more instruction sets for operating the device include values or states of one or more registers or elements of a processor circuit. In further embodiments, the first one or more instruction sets for operating the device include one or more inputs into a logic circuit. In further embodiments, the first one or more instruction sets for operating the device include one or more outputs from a logic circuit. In further embodiments, the first one or more instruction sets for operating the device include one or more instruction sets for operating an application or an object of the application.

In certain embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device as they are executed by a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first

one or more instruction sets for operating the device from a register or an element of a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from an element that is part of, operating on, or coupled to a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from at least one of: the memory unit, the device, a virtual machine, a runtime engine, a hard drive, a storage device, a peripheral device, a network connected device, or a user. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from a plurality of processor circuits, applications, memory units, devices, virtual machines, runtime engines, hard drives, storage devices, peripheral devices, network connected devices, or users.

In some embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from a logic circuit. In further embodiments, the logic circuit includes a microcontroller. In further embodiments, the receiving the first one or more instruction sets for operating the device from the logic circuit includes receiving the first one or more instruction sets for operating the device from an element of the logic circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device from the logic circuit includes receiving one or more inputs into the logic circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device from the logic circuit includes receiving one or more outputs from the logic circuit.

In certain embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from an application for operating the device. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from an application, the application including instruction sets for operating the device. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of an element that is part of, operating on, or coupled to a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a register of a processor circuit, the memory unit, a storage, or a repository where the first one or more instruction sets for operating the device are stored. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a processor circuit, the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a processing element. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a processor circuit or tracing, profiling, or instrumentation of a component of a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of an application or an object of the application. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device by an interface. The interface may include an acquisition interface.

In some embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device includes storing the first collection of object representations

correlated with the first one or more instruction sets for operating the device into a memory unit, the first collection of object representations correlated with the first one or more instruction sets for operating the device being part of a plurality of collections of object representations correlated with one or more instruction sets for operating the device stored in the memory unit.

In certain embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes executing the first one or more instruction sets for operating the device correlated with the first collection of object representations instead of or prior to an instruction set that would have been executed next. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying a register or an element of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first collection of object representations into a register or an element of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting a processor circuit to the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting a processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes transmitting, to a processor circuit for execution, the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes issuing an interrupt to a processor circuit and executing the first one or more instruction sets for operating the device correlated with the first collection of object representations following the interrupt. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying an element that is part of, operating on, or coupled to a processor circuit.

In some embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes executing, by a logic circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the logic circuit includes a microcontroller. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying an element of the logic circuit. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first collection of object representations into an element of the logic circuit. In further embodiments, the executing, by

the logic circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting the logic circuit to the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations includes replacing inputs into the logic circuit with the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations includes replacing outputs from the logic circuit with the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In certain embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes executing, by an application for operating the device, the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying an application, the application including instruction sets for operating the device. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting an application to the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting an application to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying at least one of: the memory unit, a register of a processor circuit, a storage, or a repository where instruction sets are stored or used. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets for operating an application or an object of the application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying at least one of: an element of a processor circuit, an element of the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more code segments, lines of

code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing an assembly language. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes adding or inserting additional code into a code of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes at least one of: modifying, removing, rewriting, or overwriting a code of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of an application. The branching or redirecting the code may include inserting at least one of: a branch, a jump, or a means for redirecting an execution. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes implementing a user's knowledge, style, or methodology of operating the device in a circumstance. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes executing the first one or more instruction sets for operating the device correlated with the first collection of object representations via an interface. The interface may include a modification interface.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving at least one extra information. In further embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: learning the first collection of object representations correlated with the at least one extra information.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: presenting, via a user interface, a user with an option to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving, via a user interface, a user's selection to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: rating the executed first one or more instruction sets for operating the device correlated with the first collection of object representations.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: presenting, via a user interface, a user with an option to cancel the execution of the executed first one or more instruction sets for operating the device correlated with the first collection of object representations.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving, via an input device, a user's operating directions, the user's operating directions for instructing a processor circuit on how to operate the device.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving a second collection of object representations, the second collection of object representations including one or more representations of objects detected by the sensor; receiving a second one or more instruction sets for operating the device; and learning the second collection of object representations correlated with the second one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving, by a first processor circuit of the one or more processor circuits, a first collection of object representations, the first collection of object representations including one or more representations of objects detected by a sensor. The operations may further comprise: receiving, by the first processor circuit of the one or more processor circuits, a first one or more instruction sets for operating a device. The operations may further comprise: learning, by the first processor circuit of the one or more processor circuits, the first collection of object representations correlated with the first one or more instruction sets for operating the device. The operations may further comprise: receiving, by the first processor circuit of the one or more processor circuits, a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The operations may further comprise: anticipating, by the first processor circuit of the one or more processor circuits, the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further comprise: causing, by the first processor circuit of the one or more processor circuits, an execution, by a second processor circuit of the one or more processor circuits, of the first one or more instruction sets for operating the device correlated with the first collection of object representations, the causing performed in response to the anticipating, by the first processor circuit of the one or more processor circuits, the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least

a partial match between the new collection of object representations and the first collection of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations by the device performed in response to the execution.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a first processor circuit, the first collection of object representations including one or more representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more instruction sets for operating a device by the first processor circuit. The method may further comprise: (c) learning the first collection of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the first processor circuit. The method may further comprise: (d) receiving a new collection of object representations by the first processor circuit, the new collection of object representations including one or more representations of objects detected by the sensor. The method may further comprise: (e) anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (e) performed by the first processor circuit. The method may further comprise: (f) executing, by a second processor circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations, the executing of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations by the device performed in response to the executing of (f).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for learning a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first collection of object representations, the first collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: learn the first collection of object representations correlated with the first one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first collection of object representations, the first collection of object representations including one or more representations of objects detected by a sensor. The operations may further comprise: receiving a first one or more instruction sets for

operating a device. The operations may further comprise: learning the first collection of object representations correlated with the first one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a processor circuit, the first collection of object representations including one or more representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more instruction sets for operating a device by the processor circuit. The method may further comprise: (c) learning the first collection of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the processor circuit.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: access the memory unit that comprises a plurality of collections of object representations correlated with one or more instruction sets for operating the device, the plurality of collections of object representations correlated with one or more instruction sets for operating the device including a first collection of object representations correlated with a first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: receive a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations performed in response to the executing by the processor circuit.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: accessing a memory unit that comprises a plurality of collections of object representations correlated with one or more instruction sets for operating a device, the plurality of collections of object representations correlated with one or more instruction sets for operating the device including a first collection of object representations correlated with a first one or more instruction sets for operating the device. The operations may further comprise: receiving a new collection of object representations, the new collection of object representations including one or more representations of objects detected by a sensor. The

operations may further comprise: anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further comprise: causing an execution of the first one or more instruction sets for operating the device correlated with the first collection of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations by the device performed in response to the execution.

In some aspects, the disclosure relates to a method comprising: (a) accessing a memory unit that comprises a plurality of collections of object representations correlated with one or more instruction sets for operating a device, the plurality of collections of object representations correlated with one or more instruction sets for operating the device including a first collection of object representations correlated with a first one or more instruction sets for operating the device, the accessing of (a) performed by a processor circuit. The method may further comprise: (b) receiving a new collection of object representations by the processor circuit, the new collection of object representations including one or more representations of objects detected by a sensor. The method may further comprise: (c) anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (c) performed by the processor circuit. The method may further comprise: (d) executing the first one or more instruction sets for operating the device correlated with the first collection of object representations, the executing of (d) performed in response to the anticipating of (c). The method may further comprise: (e) performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations by the device performed in response to the executing of (d).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for learning and using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first stream of collections of object representations, the first stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: learn the first stream of collections of object representations correlated with the first one or more instruction sets for operating the

device. The artificial intelligence unit may be further configured to: receive a new stream of collections of object representations, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations performed in response to the executing by the processor circuit.

In certain embodiments, each collection of object representations includes one or more representations of objects detected by the sensor at a time. In further embodiments, each collection of object representations includes one or more of object representations. In further embodiments, each collection of object representations includes or is associated with a time stamp, an order, or a time related information. In further embodiments, the first stream of collections of object representations includes one or more collections of representations of objects detected by the sensor over time. In further embodiments, the new stream of collections of object representations includes one or more collections of representations of objects detected by the sensor over time. In further embodiments, the first or the new stream of collections of object representations includes one or more collections of representations of objects in the device's surrounding. In further embodiments, the first or the new stream of collections of object representations includes one or more collections of representations of objects in a remote device's surrounding. In further embodiments, an object representation of a stream of collections of object representations includes one or more object properties. In further embodiments, the first or the new stream of collections of object representations includes one or more object properties. In further embodiments, the first stream of collections of object representations includes a comparative stream of collections of object representations whose at least one portion can be used for comparisons with at least one portion of streams of collections of object representations subsequent to the first stream of collections of object representations, the streams of collections of object representations subsequent to the first stream of collections of object representations comprising the new stream of collections of object representations. In further embodiments, the first stream of collections of object representations includes a comparative stream of collections of object representations that can be used for comparisons with the new stream of collections of object representations. In further embodiments, the new stream of collections of object representations includes an anticipatory stream of collections of object representations whose correlated one or more instruction sets can be used for anticipation of one or more instruction sets to be executed by the processor circuit.

In some embodiments, the first one or more instruction sets for operating the device include one or more instruction sets that temporally correspond to the first stream of collections of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first stream of collections of object representations include one or more instruction sets executed at a time of generating the first stream of collections of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first stream of collections of object representations include one or more instruction sets executed prior to generating

the first stream of collections of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first stream of collections of object representations include one or more instruction sets executed within a threshold period of time prior to generating the first stream of collections of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first stream of collections of object representations include one or more instruction sets executed subsequent to generating the first stream of collections of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first stream of collections of object representations include one or more instruction sets executed within a threshold period of time subsequent to generating the first stream of collections of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first stream of collections of object representations include one or more instruction sets executed within a threshold period of time prior to generating the first stream of collections of object representations or a threshold period of time subsequent to generating the first stream of collections of object representations.

In certain embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device include a knowledge of how the device operated in a circumstance. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device are included in a neuron, a node, a vertex, or an element of a knowledgebase. In further embodiments, the knowledgebase includes a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. In further embodiments, some of the neurons, nodes, vertices, or elements are interconnected. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device are structured into a knowledge cell. In further embodiments, the knowledge cell is included in a neuron, a node, a vertex, or an element of a knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device includes correlating the first stream of collections of object representations with the first one or more instruction sets for operating the device. In further embodiments, the correlating the first stream of collections of object representations with the first one or more instruction sets for operating the device includes generating a knowledge cell, the knowledge cell comprising the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device. In further embodiments, the correlating the first stream of collections of object representations with the first one or more instruction sets for operating the device includes structuring a knowledge of how the device operated in a circumstance. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device includes learning a user's knowledge, style, or methodology of operating the device in a circumstance.

In some embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device includes storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device into the memory unit, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device being part of a plurality of streams of collections of object representations correlated with one or

more instruction sets for operating the device stored in the memory unit. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device are organized into a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, each stream of collections of object representations correlated with one or more instruction sets for operating the device of the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device is included in a neuron, a node, a vertex, or an element of a knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device include a user's knowledge, style, or methodology of operating the device in circumstances. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device are stored on a remote computing device or a remote computing system. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device include an artificial intelligence system for knowledge structuring, storing, or representation. The artificial intelligence system for knowledge structuring, storing, or representation may include at least one of: a deep learning system, a supervised learning system, an unsupervised learning system, a neural network, a search-based system, an optimization-based system, a logic-based system, a fuzzy logic-based system, a tree-based system, a graph-based system, a hierarchical system, a symbolic system, a sub-symbolic system, an evolutionary system, a genetic system, a multi-agent system, a deterministic system, a probabilistic system, or a statistical system.

In certain embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes comparing at least one portion of the new stream of collections of object representations with at least one portion of the first stream of collections of object representations. In further embodiments, the at least one portion of the new stream of collections of object representations include at least one collection of object representations, at least one object representation, or at least one object property of the new stream of collections of object representations. In further embodiments, the at least one portion of the first stream of collections of object representations include at least one collection of object representations, at least one object representation, or at least one object property of the first stream of collections of object representations. In further embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes comparing at least one collection of object representations from the new stream of collections of object representations with at least one collection of object representations from the first stream of collections of object representations. In further

embodiments, the comparing at least one collection of object representations from the new stream of collections of object representations with at least one collection of object representations from the first stream of collections of object representations includes comparing at least one object representation of the at least one collection of object representations from the new stream of collections of object representations with at least one object representation of the at least one collection of object representations from the first stream of collections of object representations. The comparing at least one object representation of the at least one collection of object representations from the new stream of collections of object representations with at least one object representation of the at least one collection of object representations from the first stream of collections of object representations may include comparing at least one object property of the at least one object representation of the at least one collection of object representations from the new stream of collections of object representations with at least one object property of the at least one object representation of the at least one collection of object representations from the first stream of collections of object representations.

In some embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that there is at least a partial match between one or more portions of the new stream of collections of object representations and one or more portions of the first stream of collections of object representations. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that a similarity between at least one portion of the new stream of collections of object representations and at least one portion of the first stream of collections of object representations exceeds a similarity threshold. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining a substantial similarity between at least one portion of the new stream of collections of object representations and at least one portion of the first stream of collections of object representations. The substantial similarity may be achieved when a similarity between the at least one portion of the new stream of collections of object representations and the at least one portion of the first stream of collections of object representations exceeds a similarity threshold. The substantial similarity may be achieved when a number or a percentage of matching or partially matching portions of the new stream of collections of object representations and portions of the first stream of collections of object representations exceeds a threshold number or threshold percentage. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that a number or a percentage of matching or partially matching collections of object representations from the new stream of collections of object representations and from the first stream of collections of object representations exceeds a threshold number or threshold percentage. The matching or partially matching collections of object representations from the new stream of collections of object representations and from the first stream of collections of object representations

may be determined factoring in at least one of: an importance of a collection of object representations, an order of a collection of object representations, a threshold for a similarity in a collection of object representations, or a threshold for a difference in a collection of object representations. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that a number or a percentage of matching or partially matching object representations from the new stream of collections of object representations and from the first stream of collections of object representations exceeds a threshold number or threshold percentage. The matching or partially matching object representations from the new stream of collections of object representations and from the first stream of collections of object representations may be determined factoring in at least one of: an association of an object representation with a collection of object representations, a type of an object representation, an importance of an object representation, a threshold for a similarity in an object representation, or a threshold for a difference in an object representation. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that a number or a percentage of matching or partially matching object properties from the new stream of collections of object representations and from the first stream of collections of object representations exceeds a threshold number or threshold percentage. The matching or partially matching object properties from the new stream of collections of object representations and from the first stream of collections of object representations may be determined factoring in at least one of: an association of an object property with an object representation, an association of an object property with a collection of object representations, a category of an object property, an importance of an object property, a threshold for a similarity in an object property, or a threshold for a difference in an object property. In further embodiments, determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that there is at least a partial match between at least one collection of object representations from the new stream of collections of object representations and at least one collection of object representations from the first stream of collections of object representations. The determining that there is at least a partial match between at least one collection of object representations from the new stream of collections of object representations and at least one collection of object representations from the first stream of collections of object representations may include determining that there is at least a partial match between at least one object representation of the at least one collection of object representations from the new stream of collections of object representations and at least one object representation of the at least one collection of object representations from the first stream of collections of object representations. The determining that there is at least a partial match between at least one object representation of the at least one collection of object representations from the new stream of collections of object representations and at least one object representation of the at least one collection of object representations from the first stream of collections of object representations may include determining that there is at least a partial match between at least one object property of the at least one object representation of the at least one collection of object representations from the new stream of collections of object representations and at least one object property of the at least one object representation of the at least one collection of object representations from the first stream of collections of object representations.

In some embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations instead of or prior to an instruction set that would have been executed next. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations into a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting the processor circuit to the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting the processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes transmitting, to the processor circuit for execution, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes issuing an interrupt to the processor circuit and executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations following the interrupt. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying an element that is part of, operating on, or coupled to the processor circuit.

In certain embodiments, the processor circuit includes a logic circuit, and wherein the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the logic circuit includes a microcontroller. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying an element of the logic circuit. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes inserting the first one or more instruction sets for operating the device

correlated with the first stream of collections of object representations into an element of the logic circuit. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting the logic circuit to the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes replacing inputs into the logic circuit with the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes replacing outputs from the logic circuit with the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In some embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes causing an application for operating the device to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the application running on the processor circuit.

In certain embodiments, the system further comprises: an application including instruction sets for operating the device, the application running on the processor circuit, wherein the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying the application.

In some embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting an application to the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting an application to one or more alternate instruction sets, the application running on the processor circuit, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying at least one of: the memory unit, a register of the processor circuit, a storage, or a repository where instruction sets are stored or used. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of

object representations includes modifying one or more instruction sets for operating an application or an object of the application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying at least one of: an element of the processor circuit, an element of the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing an assembly language. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes adding or inserting additional code into a code of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object

representations includes at least one of: modifying, removing, rewriting, or overwriting a code of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of an application, the application running on the processor circuit. The branching or redirecting the code may include inserting at least one of: a branch, a jump, or a means for redirecting an execution. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes implementing a user's knowledge, style, or methodology of operating the device in a circumstance.

In certain embodiments, the system further comprises: an interface configured to cause execution of instruction sets, wherein the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations is caused by the interface. The interface may include a modification interface.

In some embodiments, the performing the one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes implementing a user's knowledge, style, or methodology of operating the device in a circumstance.

In certain embodiments, the system further comprises: an application running on the processor circuit.

In some embodiments, the instruction sets for operating the device are part of an application for operating the device, the application running on the processor circuit.

In certain embodiments, the system further comprises: an application for operating the device, the application running on the processor circuit. The application for operating the device may include the instruction sets for operating the device.

In some embodiments, the artificial intelligence unit is further configured to: receive at least one extra information. In further embodiments, the at least one extra information include one or more of: a time information, a location information, a computed information, or a contextual information. In further embodiments, the at least one extra information include one or more of: an information on an object, an information on an object representation, an information on a collection of object representations, an information on a stream of collections of object representations, an information on a device's circumstance, an information on an instruction set, an information on an application, an information on the processor circuit, an information on the device, or an information on an user. In further embodiments, the artificial intelligence unit is further configured to: learn the first stream of collections of object representations correlated with the at least one extra information. The learning the first stream of collections of object representations correlated with at least one extra information may include correlating the first stream of collections of object representations with the at least one extra information. The learning the first stream of collections of object representations correlated with at least one extra information may include storing the first stream of collections of object representations correlated with the at least one extra information into the memory unit. In further embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes anticipating the first one or more instruction sets for operating the device correlated with the first stream of

collections of object representations based on at least a partial match between an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations. The anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations may include comparing an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations. The anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations may include determining that a similarity between an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations exceeds a similarity threshold.

In certain embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: cause the user interface to present a user with an option to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In some embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: receive, via the user interface, a user's selection to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In certain embodiments, the artificial intelligence unit is further configured to: rate the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. The rating the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations may include causing a user interface to display the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations along with one or more rating values as options to be selected by a user. The rating the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations may include rating the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations without a user input.

In some embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: cause the user interface to present a user with an option to cancel the execution of the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the canceling the execution of the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes restoring the processor circuit or the device to a prior state. The restoring the processor circuit or the device to a prior state may include saving the state of the processor circuit or the device prior to executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In certain embodiments, the system further comprises: an input device configured to receive a user's operating directions, the user's operating directions for instructing the processor circuit on how to operate the device.

In some embodiments, the autonomous device operating includes a partially or a fully autonomous device operating. The partially autonomous device operating may include executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations responsive to a user confirmation. The fully autonomous device operating may include executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations without a user confirmation.

In certain embodiments, the artificial intelligence unit is further configured to: receive a second stream of collections of object representations, the second stream of collections of object representations including one or more collections of representations of objects detected by the sensor; receive a second one or more instruction sets for operating the device; and learn the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device. In further embodiments, the second stream of collections of object representations includes one or more collections of representations of objects detected by the sensor over time. In further embodiments, the second stream of collections of object representations includes or is associated with a time stamp, an order, or a time related information. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device include creating a connection between the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device. The connection may include or is associated with at least one of: an occurrence count, a weight, a parameter, or a data. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device include updating a connection between the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device. In further embodiments, the updating the connection between the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device includes updating at least one of: an occurrence count, a weight, a parameter, or a data included in or associated with the connection. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device includes storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device into a first node of a knowledgebase, and wherein the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device includes storing the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device into a second node of the knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. In further embodiments, the knowledgebase may be stored in the memory unit. The learning the first stream of collections of

object representations correlated with the first one or more instruction sets for operating the device and the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device may include creating a connection between the first node and the second node. The learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device may include updating a connection between the first node and the second node. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device is stored into a first node of a neural network and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device is stored into a second node of the neural network. The first node and the second node may be connected by a connection. The first node may be part of a first layer of the neural network and the second node may be part of a second layer of the neural network. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device is stored into a first node of a graph and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device is stored into a second node of the graph. The first node and the second node may be connected by a connection. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device is stored into a first node of a sequence and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device is stored into a second node of the sequence.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first stream of collections of object representations, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The operations may further comprise: receiving a first one or more instruction sets for operating a device. The operations may further comprise: learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device. The operations may further comprise: receiving a new stream of collections of object representations, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The operations may further comprise: anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The operations may further comprise: causing an execution of the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device

correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the execution.

In certain embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from the one or more processor circuits or from another one or more processor circuits. In further embodiments, the execution of the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations is performed by the one or more processor circuits or by another one or more processor circuits.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first stream of collections of object representations by a processor circuit, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more instruction sets for operating a device by the processor circuit. The method may further comprise: (c) learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the processor circuit. The method may further comprise: (d) receiving a new stream of collections of object representations by the processor circuit, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The method may further comprise: (e) anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, the anticipating of (e) performed by the processor circuit. The method may further comprise: (f) executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the executing of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the executing of (f).

In some embodiments, the receiving of (b) includes receiving the first one or more instruction sets for operating the device from the processor circuit or from another processor circuit. In further embodiments, the executing of (f) is performed by the processor circuit or by another processor circuit.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable as well as the following embodiments.

In certain embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device includes storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device into a memory unit, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device being part of a plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device stored in the memory unit.

In some embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations instead of or prior to an instruction set that would have been executed next. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying a register or an element of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations into a register or an element of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting a processor circuit to the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting a processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes transmitting, to a processor circuit for execution, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes issuing an interrupt to a processor circuit and executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations following the interrupt. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying an element that is part of, operating on, or coupled to a processor circuit.

In certain embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes executing, by a logic circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the logic circuit includes a microcontroller. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying an element of the logic circuit. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations into an element of the logic circuit. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting the logic circuit to the first one or more instruction sets for operating the device correlated with

the first stream of collections of object representations. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes replacing inputs into the logic circuit with the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes replacing outputs from the logic circuit with the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In some embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes executing, by an application for operating the device, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying an application, the application including instruction sets for operating the device. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting an application to the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting an application to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying at least one of: the memory unit, a register of a processor circuit, a storage, or a repository where instruction sets are stored or used. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets for operating an application or an object of the application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying at least one of: an element of a processor circuit, an element of the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further

embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing an assembly language. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes adding or inserting additional code into a code of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes at least one of: modifying, removing, rewriting, or overwriting a code of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of an application. The branching or redirecting the code may include inserting at least one of: a branch, a jump, or a means for redirecting an execution. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes implementing a user's knowledge, style, or methodology of operating the device in a circumstance. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations via an interface. The interface may include a modification interface.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving at least one extra information. In further embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: learning the first stream of collections of object representations correlated with the at least one extra information.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: presenting, via a user interface, a user with an option to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving, via a user interface, a user's selection to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: rating the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: presenting, via a user interface, a user with an option to cancel the execution of the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving, via an input device, a user's operating directions, the user's operating directions for instructing a processor circuit on how to operate the device.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving a second stream of collections of object representations, the second stream of collections of object representations including one or more collections of representations of objects detected by the sensor; receiving a second one or more instruction sets for operating the device; and learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving, by a first processor circuit of the one or more processor circuits, a first stream of collections of object representations, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The operations may further comprise: receiving, by the first processor circuit of the one or more processor circuits, a first one or more instruction sets for operating a device. The operations may further comprise: learning, by the first processor circuit of the one or more processor circuits, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device. The operations may further comprise: receiving, by the first processor circuit of the one or more processor circuits, a new stream of collections of object representations, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The operations may further comprise: anticipating, by the first processor circuit of the one or more processor circuits, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The operations may further comprise: causing, by the first processor circuit of the one or more processor circuits, an execution, by a second processor circuit of the one or more processor circuits, of the first one or more instruction sets for operating the device correlated with the first stream of collections of object

representations, the causing performed in response to the anticipating, by the first processor circuit of the one or more processor circuits, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the execution.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first stream of collections of object representations by a first processor circuit, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more instruction sets for operating a device by the first processor circuit. The method may further comprise: (c) learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the first processor circuit. The method may further comprise: (d) receiving a new stream of collections of object representations by the first processor circuit, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The method may further comprise: (e) anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, the anticipating of (e) performed by the first processor circuit. The method may further comprise: (f) executing, by a second processor circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the executing of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the executing of (f).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for learning a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first stream of collections of object representations, the first stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: learn the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first stream of collections of object representations, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The operations may further comprise: receiving a first one or more instruction sets for operating a device. The operations may further comprise: learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a non method comprising: (a) receiving a first stream of collections of object representations by a processor circuit, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more instruction sets for operating a device by the processor circuit. The method may further comprise: (c) learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the processor circuit.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: access the memory unit that comprises a plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device including a first stream of collections of object representations correlated with a first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: receive a new stream of collections of object representations, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations performed in response to the executing by the processor circuit.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits

cause the one or more processor circuits to perform operations comprising: accessing a memory unit that comprises a plurality of streams of collections of object representations correlated with one or more instruction sets for operating a device, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device including a first stream of collections of object representations correlated with a first one or more instruction sets for operating the device. The operations may further comprise: receiving a new stream of collections of object representations, the new stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The operations may further comprise: anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The operations may further comprise: causing an execution of the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the execution.

In some aspects, the disclosure relates to a method comprising: (a) accessing a memory unit that comprises a plurality of streams of collections of object representations correlated with one or more instruction sets for operating a device, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device including a first stream of collections of object representations correlated with a first one or more instruction sets for operating the device, the accessing of (a) performed by a processor circuit. The method may further comprise: (b) receiving a new stream of collections of object representations by the processor circuit, the new stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The method may further comprise: (c) anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, the anticipating of (c) performed by the processor circuit. The method may further comprise: (d) executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the executing of (d) performed in response to the anticipating of (c). The method may further comprise: (e) performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the executing of (d).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for learning and using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices.

In some embodiments, the system comprises: a logic circuit configured to receive inputs and produce outputs, wherein the outputs are used for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first collection of object representations, the first collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more inputs, wherein the first one or more inputs are also received by the logic circuit. The artificial intelligence unit may be further configured to: learn the first collection of object representations correlated with the first one or more inputs. The artificial intelligence unit may be further configured to: receive a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may be further configured to: cause the logic circuit to receive the first one or more inputs correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by one or more outputs for operating the device produced by the logic circuit.

In some embodiments, the logic circuit configured to receive inputs and produce outputs includes a logic circuit configured to produce outputs based at least in part on logic operations performed on the inputs. In further embodiments, the learning the first collection of object representations correlated with the first one or more inputs includes correlating the first collection of object representations with the first one or more inputs. In further embodiments, the learning the first collection of object representations correlated with the first one or more inputs includes storing the first collection of object representations correlated with the first one or more inputs into the memory unit, the first collection of object representations correlated with the first one or more inputs being part of a plurality of collections of object representations correlated with one or more inputs stored in the memory unit. In further embodiments, the anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes comparing at least one portion of the new collection of object representations with at least one portion of the first collection of object representations. In further embodiments, the anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes determining that there is at least a partial match between the new collection of object representations and the first collection of object representations. In further embodiments, the causing the logic circuit to receive the first one or more inputs correlated with the first collection of object representations includes transmitting, to the logic circuit, the first one or more inputs correlated with the first collection of object representations. In further embodiments, the causing the logic circuit to receive the first one or more inputs correlated with the first collection of object representations includes replacing one or more inputs into the logic circuit with the first one or more inputs correlated with the first collection of object representations.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first collection of object representations, the first collection of object representations including one or more representations of objects detected by a sensor. The operations may further comprise: receiving a first one or more inputs, wherein the first one or more inputs are also received by a logic circuit, and wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device. The operations may further comprise: learning the first collection of object representations correlated with the first one or more inputs. The operations may further comprise: receiving a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The operations may further comprise: anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further comprise: causing the logic circuit to receive the first one or more inputs correlated with the first collection of object representations, the causing performed in response to the anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, wherein the device performs one or more operations defined by one or more outputs for operating the device produced by the logic circuit.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a processor circuit, the first collection of object representations including one or more representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more inputs by the processor circuit, wherein the first one or more inputs are also received by a logic circuit, and wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device. The method may further comprise: (c) learning the first collection of object representations correlated with the first one or more inputs, the learning of (c) performed by the processor circuit. The method may further comprise: (d) receiving a new collection of object representations by the processor circuit, the new collection of object representations including one or more representations of objects detected by the sensor. The method may further comprise: (e) anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (e) performed by the processor circuit. The method may further comprise: (f) receiving, by the logic circuit, the first one or more inputs correlated with the first collection of object representations, the receiving of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the device, one or more operations defined by one or more outputs for operating the device produced by the logic circuit.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for learning and using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a logic circuit configured to receive inputs and produce outputs, wherein the outputs are used for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence. The artificial intelligence unit may be configured to: receive a first collection of object representations, the first collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more outputs, the first one or more outputs transmitted from the logic circuit. The artificial intelligence unit may be further configured to: learn the first collection of object representations correlated with the first one or more outputs. The artificial intelligence unit may be further configured to: receive a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more outputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may be further configured to: cause the device to perform one or more operations defined by the first one or more outputs correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit.

In some embodiments, the learning the first collection of object representations correlated with the first one or more outputs includes correlating the first collection of object representations with the first one or more outputs. In further embodiments, the learning the first collection of object representations correlated with the first one or more outputs includes storing the first collection of object representations correlated with the first one or more outputs into the memory unit, the first collection of object representations correlated with the first one or more outputs being part of a plurality of collections of object representations correlated with one or more outputs stored in the memory unit. In further embodiments, the anticipating the first one or more outputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes comparing at least one portion of the new collection of object representations with at least one portion of the first collection of object representations. In further embodiments, the anticipating the first one or more outputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes determining that there is at least a partial match between the new collection of object representations and the first collection of object representations. In further embodiments, the causing the device to perform one or more operations defined by the first one or more outputs correlated with the first collection of object representations includes replacing one or more outputs from the logic circuit with the first one or more outputs correlated with the first collection of object representations.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first collection of object representations, the first collection of object representations including one or more representations of objects

detected by a sensor. The operations may further comprise: receiving a first one or more outputs, the first one or more outputs transmitted from a logic circuit, wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device. The operations may further comprise: learning the first collection of object representations correlated with the first one or more outputs. The operations may further comprise: receiving a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The operations may further comprise: anticipating the first one or more outputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further comprise: causing the device to perform one or more operations defined by the first one or more outputs correlated with the first collection of object representations, the causing performed in response to the anticipating the first one or more outputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a processor circuit, the first collection of object representations including one or more representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more outputs by the processor circuit, the first one or more outputs transmitted from a logic circuit, wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device. The method may further comprise: (c) learning the first collection of object representations correlated with the first one or more outputs, the learning of (c) performed by the processor circuit. The method may further comprise: (d) receiving a new collection of object representations by the processor circuit, the new collection of object representations including one or more representations of objects detected by the sensor. The method may further comprise: (e) anticipating the first one or more outputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (e) performed by the processor circuit. The method may further comprise: (f) performing, by the device, one or more operations defined by the first one or more outputs correlated with the first collection of object representations, the one or more operations by the device performed in response to the anticipating of (e).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for learning and using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: an actuator configured to receive inputs and perform motions. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first collection of object representations, the first collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence

unit may be further configured to: receive a first one or more inputs, wherein the first one or more inputs are also received by the actuator. The artificial intelligence unit may be further configured to: learn the first collection of object representations correlated with the first one or more inputs. The artificial intelligence unit may be further configured to: receive a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may be further configured to: cause the actuator to receive the first one or more inputs correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the actuator performs one or more motions defined by the first one or more inputs correlated with the first collection of object representations.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first collection of object representations, the first collection of object representations including one or more representations of objects detected by a sensor. The operations may further comprise: receiving a first one or more inputs, wherein the first one or more inputs are also received by an actuator, and wherein the actuator is configured to receive inputs and perform motions. The operations may further comprise: learning the first collection of object representations correlated with the first one or more inputs. The operations may further comprise: receiving a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The operations may further comprise: anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further comprise: causing the actuator to receive the first one or more inputs correlated with the first collection of object representations, the causing performed in response to the anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, wherein the actuator performs one or more motions defined by the first one or more inputs correlated with the first collection of object representations.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a processor circuit, the first collection of object representations including one or more representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more inputs by the processor circuit, wherein the first one or more inputs are also received by an actuator, and wherein the actuator is configured to receive inputs and perform motions. The method may further comprise: (c) learning the first collection of object representations correlated with the first one or more inputs, the learning of (c) performed by the processor circuit. The method may further comprise: (d) receiving a new collection of object representations by the processor circuit, the new collection of object representations including one or more representations of objects detected by the sensor. The method may further comprise: (e) anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (e) performed by the

processor circuit. The method may further comprise: (f) receiving, by the actuator, the first one or more inputs correlated with the first collection of object representations, the receiving of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the actuator, one or more motions defined by the first one or more inputs correlated with the first collection of object representations.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

Other features and advantages of the disclosure will become apparent from the following description, including the claims and drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

Fig. 1 illustrates a block diagram of Computing Device 70 that can provide processing capabilities used in some of the disclosed embodiments.

Fig. 2 illustrates an embodiment of Device 98 comprising Unit for Learning and/or Using a Device's Circumstances for Autonomous Device Operation (DCADO Unit 100).

Figs. 3A-3E illustrate various embodiments of Sensors 92 and elements of Object Processing Unit 93.

Figs. 4A-4B, illustrate an exemplary embodiment of Objects 615 detected in Device's 98 surrounding, and resulting Collection of Object Representations 525.

Fig. 5 illustrates some embodiments of obtaining instruction sets, data, and/or other information through tracing, profiling, or sampling of Processor 11 registers, memory, or other computing system components.

Figs. 6A-6B illustrate some embodiments of obtaining instruction sets, data, and/or other information through tracing, profiling, or sampling of Logic Circuit 250.

Figs. 7A-7E illustrate some embodiments of Instruction Sets 526.

Figs. 8A-8B illustrate some embodiments of Extra Information 527.

Fig. 9 illustrates an embodiment where DCADO Unit 100 is part of or operating on Processor 11.

Fig. 10 illustrates an embodiment where DCADO Unit 100 resides on Server 96 accessible over Network 95.

Fig. 11 illustrates an embodiment of learning and/or using Remote Device's 97 circumstances for autonomous Remote Device 97 operation.

Fig. 12 illustrates an embodiment of Artificial Intelligence Unit 110.

Fig. 13 illustrates an embodiment of Knowledge Structuring Unit 520 correlating individual Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527.

Fig. 14 illustrates another embodiment of Knowledge Structuring Unit 520 correlating individual Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527.

Fig. 15 illustrates an embodiment of Knowledge Structuring Unit 520 correlating streams of Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527.

Fig. 16 illustrates another embodiment of Knowledge Structuring Unit 520 correlating streams of Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527.

Fig. 17 illustrates various artificial intelligence methods, systems, and/or models that can be utilized in DCADO Unit 100 embodiments.

Fig. 18A-18C illustrate embodiments of interconnected Knowledge Cells 800 and updating weights of Connections 853.

Fig. 19 illustrates an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Collection of Knowledge Cells 530d.

Fig. 20 illustrates an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Neural Network 530a.

Fig. 21 illustrates an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Neural Network 530a comprising shortcut Connections 853.

Fig. 22 illustrates an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Graph 530b.

Fig. 23 illustrates an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Collection of Sequences 530c.

Fig. 24 illustrates an embodiment of determining anticipatory Instruction Sets 526 from a single Knowledge Cell 800.

Fig. 25 illustrates an embodiment of determining anticipatory Instruction Sets 526 by traversing a single Knowledge Cell 800.

Fig. 26 illustrates an embodiment of determining anticipatory Instruction Sets 526 using collective similarity comparisons.

Fig. 27 illustrates an embodiment of determining anticipatory Instruction Sets 526 using Neural Network 530a.

Fig. 28 illustrates an embodiment of determining anticipatory Instruction Sets 526 using Graph 530b.

Fig. 29 illustrates an embodiment of determining anticipatory Instruction Sets 526 using Collection of Sequences 530c.

Fig. 30 illustrates some embodiments of modifying execution and/or functionality of Processor 11 through modification of Processor 11 registers, memory, or other computing system components.

Figs. 31A-31B illustrate some embodiments of modifying execution and/or functionality of Logic Circuit 250 through modification of inputs and/or outputs of Logic Circuit 250.

Fig. 32 illustrates a flow chart diagram of an embodiment of method 9100 for learning and/or using a device's circumstances for autonomous device operation.

Fig. 33 illustrates a flow chart diagram of an embodiment of method 9200 for learning and/or using a device's circumstances for autonomous device operation.

Fig. 34 illustrates a flow chart diagram of an embodiment of method 9300 for learning and/or using a device's circumstances for autonomous device operation.

Fig. 35 illustrates a flow chart diagram of an embodiment of method 9400 for learning and/or using a device's circumstances for autonomous device operation.

Fig. 36 illustrates a flow chart diagram of an embodiment of method 9500 for learning and/or using a device's circumstances for autonomous device operation.

Fig. 37 illustrates a flow chart diagram of an embodiment of method 9600 for learning and/or using a device's circumstances for autonomous device operation.

Fig. 38 illustrates an exemplary embodiment of Loader 98a.

Fig. 39 illustrates an exemplary embodiment of Boat 98b.

Fig. 40 illustrates an exemplary embodiment of utilizing Area of Interest 450 around Boat 98b.

Like reference numerals in different figures indicate like elements. Horizontal or vertical "..." or other such indicia may be used to indicate additional instances of the same type of element. n, m, x, or other such letters or indicia represent integers or other sequential numbers that follow the sequence where they are indicated. It should be noted that n, m, x, or other such letters or indicia may represent different numbers in different elements even where the elements are depicted in the same figure. In general, n, m, x, or other such letters or indicia may follow the sequence and/or context where they are indicated. Any of these or other such letters or indicia may be used interchangeably depending on context and space available. The drawings are not necessarily to scale, with emphasis instead being placed upon illustrating the embodiments, principles, and concepts of the disclosure. A line or arrow between any of the disclosed elements comprises an interface that enables the coupling, connection, and/or interaction between the elements.

DETAILED DESCRIPTION

The disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation comprise apparatuses, systems, methods, features, functionalities, and/or applications that enable learning a device's circumstances including objects with various properties along with correlated instruction sets for operating the device, storing this knowledge in a knowledgebase (i.e. neural network, graph, sequences, etc.), and operating a device autonomously. The disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, any of their elements, any of their embodiments, or a combination thereof can generally be referred to as DCADO, DCADO Unit, or as other suitable name or reference.

Referring now to Fig. 1, an embodiment is illustrated of Computing Device 70 (also referred to simply as computing device, computing system, or other suitable name or reference, etc.) that can provide processing capabilities used in some embodiments of the forthcoming disclosure. Later described devices, systems, and methods, in combination with processing capabilities of Computing Device 70, enable learning and/or using a device's circumstances for autonomous device operation and/or other functionalities described herein. Various embodiments of the disclosed devices, systems, and methods include hardware, functions, logic, programs, and/or a combination thereof that can be implemented using any type or form of computing, computing enabled, or other device or system such as a mobile device, a computer, a computing enabled telephone, a server, a gaming device, a television device, a digital camera, a GPS receiver, a media player, an embedded device, a supercomputer, a wearable device, an implantable device, a cloud, or any other type or form of computing, computing enabled, or

other device or system capable of performing the operations described herein.

In some designs, Computing Device 70 comprises hardware, processing techniques or capabilities, programs, or a combination thereof. Computing Device 70 includes one or more central processing units, which may also be referred to as processors 11. Processor 11 includes one or more memory ports 10 and/or one or more input-output ports, also referred to as I/O ports 15, such as I/O ports 15A and 15B. Processor 11 may be special or general purpose. Computing Device 70 may further include memory 12, which can be connected to the remainder of the components of Computing Device 70 via bus 5. Memory 12 can be connected to processor 11 via memory port 10. Computing Device 70 may also include display device 21 such as a monitor, projector, glasses, and/or other display device. Computing Device 70 may also include Human-machine Interface 23 such as a keyboard, a pointing device, a mouse, a touchscreen, a joystick, a remote controller, and/or other input device. In some implementations, Human-machine Interface 23 can be connected with bus 5 or directly connected with specific elements of Computing Device 70. Computing Device 70 may include additional elements such as one or more input/output devices 13. Processor 11 may include or be interfaced with cache memory 14. Storage 27 may include memory, which provides an operating system 17 (i.e. also referred to as OS 17, etc.), additional application programs 18, and/or data space 19 in which additional data or information can be stored. Alternative memory device 16 can be connected to the remaining components of Computing Device 70 via bus 5. Network interface 25 can also be connected with bus 5 and be used to communicate with external computing devices via a network. Some or all described elements of Computing Device 70 can be directly or operatively connected or coupled with each other using any other connection means known in art. Other additional elements may be included as needed, or some of the disclosed ones may be excluded, or a combination thereof may be utilized in alternate implementations of Computing Device 70.

Processor 11 includes one or more circuits or devices that can execute instructions fetched from memory 12 and/or other element. Processor 11 may include any combination of hardware and/or processing techniques or capabilities for executing or implementing logic functions or programs. Processor 11 may include a single core or a multi core processor. Processor 11 includes the functionality for loading operating system 17 and operating any application programs 18 thereon. In some embodiments, Processor 11 can be provided in a microprocessing or a processing unit, such as, for example, Snapdragon processor produced by Qualcomm Inc., processor by Intel Corporation of Mountain View, California, processor manufactured by Motorola Corporation of Schaumburg, Ill.; processor manufactured by Transmeta Corporation of Santa Clara, Calif.; processor manufactured by International Business Machines of White Plains, N.Y.; processor manufactured by Advanced Micro Devices of Sunnyvale, California, or any computing circuit or device for performing similar functions. In other embodiments, processor 11 can be provided in a graphics processing unit (GPU), visual processing unit (VPU), or other highly parallel processing circuit or device such as, for example, nVidia GeForce line of GPUs, AMD Radeon line of GPUs, and/or others. Such GPUs or other highly parallel processing circuits or devices may provide superior performance in processing operations on neural networks, graphs, and/or other data structures. In further embodiments, processor 11 can be provided in a micro controller such as, for example, Texas instruments, Atmel, Microchip Technology, ARM, Silicon Labs, Intel, and/or other lines of micro controllers. In further embodiments, processor 11 can be provided in a quantum processor such as, for example, D-Wave Systems, Microsoft, Intel, IBM, Google, Toshiba, and/or other lines of quantum processors. In further embodiments, processor 11 can be provided in a biocomputer

such as DNA-based computer, protein-based computer, molecule-based computer, and/or others. In further embodiments, processor 11 includes any circuit or device for performing logic operations. Processor 11 can be based on any of the aforementioned or other available processors capable of operating as described herein. Computing Device 70 may include one or more of the aforementioned or other processors. In some designs, processor 11 can communicate with memory 12 via a system bus 5. In other designs, processor 11 can communicate directly with memory 12 via a memory port 10.

Memory 12 includes one or more circuits or devices capable of storing data. In some embodiments, Memory 12 can be provided in a semiconductor or electronic memory chip such as static random access memory (SRAM), Flash memory, Burst SRAM or SynchBurst SRAM (BSRAM), Dynamic random access memory (DRAM), Fast Page Mode DRAM (FPM DRAM), Enhanced DRAM (EDRAM), Extended Data Output RAM (EDO RAM), Extended Data Output DRAM (EDO DRAM), Burst Extended Data Output DRAM (BEDO DRAM), Enhanced DRAM (EDRAM), synchronous DRAM (SDRAM), JEDEC SRAM, PC100 SDRAM, Double Data Rate SDRAM (DDR SDRAM), Enhanced SDRAM (ESDRAM), SyncLink DRAM (SLDRAM), Direct Rambus DRAM (DRDRAM), Ferroelectric RAM (FRAM), and/or others. In other embodiments, Memory 12 includes any volatile memory. In general, Memory 12 can be based on any of the aforementioned or other available memories capable of operating as described herein.

Storage 27 includes one or more devices or mediums capable of storing data. In some embodiments, Storage 27 can be provided in a device or medium such as a hard drive, flash drive, optical disk, and/or others. In other embodiments, Storage 27 can be provided in a biological storage device such as DNA-based storage device, protein-based storage device, molecule-based storage device, and/or others. In further embodiments, Storage 27 can be provided in an optical storage device such as holographic storage, and/or others. In further embodiments, Storage 27 may include any non-volatile memory. In general, Storage 27 can be based on any of the aforementioned or other available storage devices or mediums capable of operating as described herein. In some aspects, Storage 27 may include any features, functionalities, and embodiments of Memory 12, and vice versa, as applicable.

Processor 11 can communicate directly with cache memory 14 via a connection means such as a secondary bus which may also sometimes be referred to as a backside bus. In some embodiments, processor 11 can communicate with cache memory 14 using the system bus 5. Cache memory 14 may typically have a faster response time than main memory 12 and can include a type of memory which is considered faster than main memory 12 such as, for example, SRAM, BSRAM, or EDRAM. Cache memory includes any structure such as multilevel caches, for example. In some embodiments, processor 11 can communicate with one or more I/O devices 13 via a system bus 5. Various busses can be used to connect processor 11 to any of the I/O devices 13 such as a VESA VL bus, an ISA bus, an EISA bus, a MicroChannel Architecture (MCA) bus, a PCI bus, a PCI-X bus, a PCI-Express bus, a NuBus, and/or others. In some embodiments, processor 11 can communicate directly with I/O device 13 via HyperTransport, Rapid I/O, or InfiniBand. In further embodiments, local busses and direct communication can be mixed. For example, processor 11 can communicate with an I/O device 13 using a local interconnect bus and communicate with another I/O device 13 directly. Similar configurations can be used for any other components described herein.

Computing Device 70 may further include alternative memory such as a SD memory slot, a USB memory stick, an optical drive such as a CD-ROM drive, a CD-R/RW drive, a DVD-ROM drive or a BlueRay disc, a hard-drive, and/or any other device comprising non-volatile memory suitable for storing data or installing application programs. Computing Device 70 may further include a storage device 27 comprising any type or form of non-volatile memory for storing an operating system (OS) such as any type or form of Windows OS, Mac OS, Unix OS, Linux OS, Android OS, iPhone OS, mobile version of Windows OS, an embedded OS, or any other OS that can operate on Computing Device 70. Computing Device 70 may also include application programs 18, and/or data space 19 for storing additional data or information. In some embodiments, alternative memory 16 can be used as or similar to storage device 27. Additionally, OS 17 and/or application programs 18 can be operable from a bootable medium such as, for example, a flash drive, a micro SD card, a bootable CD or DVD, and/or other bootable medium.

Application Program 18 (also referred to as program, computer program, application, script, code, or other suitable name or reference) comprises instructions that can provide functionality when executed by processor 11. As such, Application Program 18 may be used to operate (i.e. perform operations on/with) or control a device or system. Application program 18 can be implemented in a high-level procedural or object-oriented programming language, or in a low-level machine or assembly language. Any language used can be compiled, interpreted, or otherwise translated into machine language. Application program 18 can be deployed in any form including as a stand-alone program or as a module, component, subroutine, or other unit suitable for use in a computing system. Application program 18 does not necessarily correspond to a file in a file system. A program can be stored in a portion of a file that may hold other programs or data, in a single file dedicated to the program, or in multiple files (i.e. files that store one or more modules, sub programs, or portions of code, etc.). Application Program 18 can be delivered in various forms such as, for example, executable file, library, script, plugin, addon, applet, interface, console application, web application, application service provider (ASP)-type application, operating system, and/or other forms. Application program 18 can be deployed to be executed on one computing device or on multiple computing devices (i.e. cloud, distributed, or parallel computing, etc.), or at one site or distributed across multiple sites interconnected by a communication network or an interface.

Network interface 25 can be utilized for interfacing Computing Device 70 with other devices via a network through a variety of connections including telephone lines, wired or wireless connections, LAN or WAN links (i.e. 802.11, T1, T3, 56 kb, X.25, etc.), broadband connections (i.e. ISDN, Frame Relay, ATM, etc.), or a combination thereof. Examples of networks include the Internet, an intranet, an extranet, a local area network (LAN), a wide area network (WAN), a personal area network (PAN), a home area network (HAN), a campus area network (CAN), a metropolitan area network (MAN), a global area network (GAN), a storage area network (SAN), virtual network, a virtual private network (VPN), a Bluetooth network, a wireless network, a wireless LAN, a radio network, a HomePNA, a power line communication network, a G.hn network, an optical fiber network, an Ethernet network, an active networking network, a client-server network, a peer-to-peer network, a bus network, a star network, a ring network, a mesh network, a star-bus network, a tree network, a hierarchical topology network, and/or other networks. Network interface 25 may include a built-in network adapter, network interface card, PCMCIA network card, card bus network adapter, wireless network adapter, Bluetooth network adapter, WiFi network adapter, USB network adapter, modem, and/or any other device suitable for interfacing Computing Device 70 with any type of network capable of communication and/or operations described herein.

I/O devices 13 may be present in various shapes or forms in Computing Device 70. Examples of I/O device 13 capable of input include a joystick, a keyboard, a mouse, a trackpad, a trackpoint, a touchscreen, a trackball, a microphone, a drawing tablet, a glove, a tactile input device, a still or video camera, and/or other input device. Examples of I/O device 13 capable of output include a video display, a touchscreen, a projector, a glasses, a speaker, a tactile output device, and/or other output device. Examples of I/O device 13 capable of input and output include a disk drive, an optical storage device, a modem, a network card, and/or other input/output device. I/O device 13 can be interfaced with processor 11 via an I/O port 15, for example. In some aspects, I/O device 13 can be a bridge between system bus 5 and an external communication bus such as a USB bus, an Apple Desktop Bus, an RS-232 serial connection, a SCSI bus, a FireWire bus, a FireWire 800 bus, an Ethernet bus, an AppleTalk bus, a Gigabit Ethernet bus, an Asynchronous Transfer Mode bus, a HIPPI bus, a Super HIPPI bus, a SerialPlus bus, a SCI/LAMP bus, a FibreChannel bus, a Serial Attached small computer system interface bus, and/or other bus.

An output interface (not shown) such as a graphical user interface, an acoustic output interface, a tactile output interface, any device driver (i.e. audio, video, or other driver), and/or other output interface or system can be utilized to process output from elements of Computing Device 70 for conveyance on an output device such as Display 21. In some aspects, Display 21 or other output device itself may include an output interface for processing output from elements of Computing Device 70. Further, an input interface (not shown) such as a keyboard listener, a touchscreen listener, a mouse listener, any device driver (i.e. audio, video, keyboard, mouse, touchscreen, or other driver), and/or other input interface or system can be utilized to process input from Human-machine Interface 23 or other input device for use by elements of Computing Device 70. In some aspects, Human-machine Interface 23 or other input device itself may include an input interface for processing input for use by elements of Computing Device 70.

Computing Device 70 may include or be connected to multiple display devices 21. Display devices 21 can each be of the same or different type or form. Computing Device 70 and/or its elements comprise any type or form of suitable hardware, programs, or a combination thereof to support, enable, or provide for the connection and use of multiple display devices 21. In one example, Computing Device 70 includes any type or form of video adapter, video card, driver, and/or library to interface, communicate, connect, or otherwise use display devices 21. In some aspects, a video adapter may include multiple connectors to interface to multiple display devices 21. In other aspects, Computing Device 70 includes multiple video adapters, with each video adapter connected to one or more display devices 21. In some embodiments, Computing Device's 70 operating system can be configured for using multiple displays 21. In other embodiments, one or more display devices 21 can be provided by one or more other computing devices such as remote computing devices connected to Computing Device 70 via a network or an interface.

Computing Device 70 can operate under the control of operating system 17, which may support Computing Device's 70 basic functions, interface with and manage hardware resources, interface with and manage peripherals, provide common services for application programs, schedule tasks, and/or perform other functionalities. A modern operating system enables features and functionalities such as a high resolution display, graphical user interface (GUI), touchscreen, cellular network connectivity (i.e. mobile operating system, etc.), Bluetooth connectivity, WiFi connectivity, global positioning system (GPS) capabilities, mobile navigation, microphone, speaker, still picture camera, video camera, voice recorder, speech recognition, music player, video player, near field communication,

personal digital assistant (PDA), and/or other features, functionalities, or applications. For example, Computing Device 70 can use any conventional operating system, any embedded operating system, any real-time operating system, any open source operating system, any video gaming operating system, any proprietary operating system, any online operating system, any operating system for mobile computing devices, or any other operating system capable of running on Computing Device 70 and performing operations described herein. Example of operating systems include Windows XP, Windows 7, Windows 8, Windows 10, etc. manufactured by Microsoft Corporation of Redmond, Wash.; Mac OS, iPhone OS, etc. manufactured by Apple Computer of Cupertino, Calif.; OS/2 manufactured by International Business Machines of Armonk, N.Y.; Linux, a freely-available operating system distributed by Caldera Corp. of Salt Lake City, Utah; or any type or form of a Unix operating system, and/or others. Any operating systems such as the ones for Android devices can similarly be utilized.

Computing Device 70 can be implemented as or be part of various model architectures such as web services, distributed computing, grid computing, cloud computing, and/or other architectures. For example, in addition to the traditional desktop, server, or mobile operating system architectures, a cloud-based operating system can be utilized to provide the structure on which embodiments of the disclosure can be implemented. Other aspects of Computing Device 70 can also be implemented in the cloud without departing from the spirit and scope of the disclosure. For example, memory, storage, processing, and/or other elements can be hosted in the cloud. In some embodiments, Computing Device 70 can be implemented on multiple devices. For example, a portion of Computing Device 70 can be implemented on a mobile device and another portion can be implemented on wearable electronics.

Computing Device 70 can be or include any mobile device, a mobile phone, a smartphone (i.e. iPhone, Windows phone, Blackberry phone, Android phone, etc.), a tablet, a personal digital assistant (PDA), wearable electronics, implantable electronics, and/or other mobile device capable of implementing the functionalities described herein. Computing Device 70 can also be or include an embedded device, which can be any device or system with a dedicated function within another device or system. Embedded systems range from the simplest ones dedicated to one task with no user interface to complex ones with advanced user interface that may resemble modern desktop computer systems. Examples of devices comprising an embedded device include a mobile telephone, a personal digital assistant (PDA), a gaming device, a media player, a digital still or video camera, a pager, a television device, a set-top box, a personal navigation device, a global positioning system (GPS) receiver, a portable storage device (i.e. a USB flash drive, etc.), a digital watch, a DVD player, a printer, a microwave oven, a washing machine, a dishwasher, a gateway, a router, a hub, an automobile entertainment system, an automobile navigation system, a refrigerator, a washing machine, a factory automation device, an assembly line device, a factory floor monitoring device, a thermostat, an automobile, a factory controller, a telephone, a network bridge, and/or other devices. An embedded device can operate under the control of an operating system for embedded devices such as MicroC/OS-II, QNX, VxWorks, eCos, TinyOS, Windows Embedded, Embedded Linux, and/or other embedded device operating systems.

Various implementations of the disclosed devices, systems, and methods can be realized in digital electronic circuitry, integrated circuitry, logic gates, application specific integrated circuits (ASICs), field programmable gate arrays (FPGAs), computer hardware, firmware, programs, virtual machines, and/or combinations thereof including their structural, logical, and/or physical equivalents.

The disclosed devices, systems, and methods may include clients and servers. A client and server are generally, but not always, remote from each other and typically, but not always, interact via a network or an interface. The relationship of a client and server may arise by virtue of computer programs running on their respective computers and having a client-server relationship to each other, for example.

The disclosed devices, systems, and methods can be implemented in a computing system that includes a back end component, a middleware component, a front end component, or any combination thereof. The components of the system can be interconnected by any form or medium of digital data communication such as, for example, a network.

Computing Device 70 may include or be interfaced with a computer program product comprising instructions or logic encoded on a computer-readable medium. Such instructions or logic, when executed, may configure or cause one or more processors to perform the operations and/or functionalities disclosed herein. For example, a computer program can be provided or encoded on a computer-readable medium such as an optical medium (i.e. DVD-ROM, etc.), flash drive, hard drive, any memory, firmware, or other medium. Computer program can be installed onto a computing device to cause the computing device to perform the operations and/or functionalities disclosed herein. Machine-readable medium, computer-readable medium, or other such terms may refer to any computer program product, apparatus, and/or device for providing instructions and/or data to one or more programmable processors. As such, machine-readable medium includes any medium that can send and/or receive machine instructions as a machine-readable signal. Examples of a machine-readable medium include a volatile and/or non-volatile medium, a removable and/or non-removable medium, a communication medium, a storage medium, and/or other medium. A communication medium, for example, can transmit computer readable instructions and/or data in a modulated data signal such as a carrier wave or other transport technique, and may include any other form of information delivery medium known in art. A non-transitory machine-readable medium comprises all machine-readable media except for a transitory, propagating signal.

In some embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, can be implemented entirely or in part in a device (i.e. microchip, circuitry, logic gates, electronic device, computing device, special or general purpose processor, etc.) or system that comprises (i.e. hard coded, internally stored, etc.) or is provided with (i.e. externally stored, etc.) instructions for implementing DCADO functionalities. As such, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, may include the processing, memory, storage, and/or other features, functionalities, and embodiments of Computing Device 70 or elements thereof. Such device or system can operate on its own (i.e. standalone device or system, etc.), be embedded in another device or system (i.e. an industrial machine, a robot, a vehicle, a toy, a smartphone, a television device, an appliance, and/or any other device or system capable of housing the elements needed for DCADO functionalities), work in combination with other devices or systems, or be available in any other configuration. In other embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, may include Alternative Memory 16 that provides instructions for implementing DCADO functionalities to one or more Processors 11. In further embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, can be

implemented entirely or in part as a computer program and executed by one or more Processors 11. Such program can be implemented in one or more modules or units of a single or multiple computer programs. Such program may be able to attach to or interface with, inspect, and/or take control of another application program to implement DCADO functionalities. In further embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, can be implemented as a network, web, distributed, cloud, or other such application accessed on one or more remote computing devices (i.e. servers, cloud, etc.) via Network Interface 25, such remote computing devices including processing capabilities and instructions for implementing DCADO functionalities. In further embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, can be (1) attached to or interfaced with any computing device or application program, (2) included as a feature of an operating system, (3) built (i.e. hard coded, etc.) into any computing device or application program, and/or (4) available in any other configuration to provide its functionalities.

In some embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, can be implemented at least in part in a computer program such as Java application or program. Java provides a robust and flexible environment for application programs including flexible user interfaces, robust security, built-in network protocols, powerful application programming interfaces, database or DBMS connectivity and interfacing functionalities, file manipulation capabilities, support for networked applications, and/or other features or functionalities. Application programs based on Java can be portable across many devices, yet leverage each device's native capabilities. Java supports the feature sets of most smartphones and a broad range of connected devices while still fitting within their resource constraints. Various Java platforms include virtual machine features comprising a runtime environment for application programs. Java platforms provide a wide range of user-level functionalities that can be implemented in application programs such as displaying text and graphics, playing and recording audio content, displaying and recording visual content, communicating with another computing device, and/or other functionalities. It should be understood that the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, are programming language, platform, and operating system independent. Examples of programming languages that can be used instead of or in addition to Java include C, C++, Cobol, Python, Java Script, Tcl, Visual Basic, Pascal, VB Script, Perl, PHP, Ruby, and/or other programming languages capable of implementing the functionalities described herein.

Where a reference to a specific file or file type is used herein, other files or file types can be substituted.

Where a reference to a data structure is used herein, it should be understood that any variety of data structures can be used such as, for example, array, list, linked list, doubly linked list, queue, tree, heap, graph, map, grid, matrix, multi-dimensional matrix, table, database, database management system (DBMS), file, neural network, and/or any other type or form of a data structure including a custom one. A data structure may include one or more fields or data fields that are part of or associated with the data structure. A field or data field may include a data, an object, a data structure, and/or any other element or a reference/pointer thereto. A data structure can be stored in one or more memories, files, or other repositories. A data structure and/or elements thereof, when stored in a memory, file, or other repository, may be stored in a different arrangement than the arrangement of the data structure and/or elements thereof. For example, a sequence of elements can be stored in an arrangement other

than a sequence in a memory, file, or other repository.

Where a reference to a repository is used herein, it should be understood that a repository may be or include one or more files or file systems, one or more storage locations or structures, one or more storage systems, one or more memory locations or structures, and/or other file, storage, memory, or data arrangements.

Where a reference to an interface is used herein, it should be understood that the interface comprises any hardware, device, system, program, method, and/or combination thereof that enable direct or operative coupling, connection, and/or interaction of the elements between which the interface is indicated. A line or arrow shown in the figures between any of the depicted elements comprises such interface. Examples of an interface include a direct connection, an operative connection, a wired connection (i.e. wire, cable, etc.), a wireless connection, a device, a network, a bus, a circuit, a firmware, a driver, a bridge, a program, a combination thereof, and/or others.

Where a reference to an element coupled or connected to another element is used herein, it should be understood that the element may be in communication or other interactive relationship with the other element. Furthermore, an element coupled or connected to another element can be coupled or connected to any other element in alternate implementations. Terms coupled, connected, interfaced, or other such terms may be used interchangeably herein depending on context.

Where a reference to an element matching another element is used herein, it should be understood that the element may be equivalent or similar to the other element. Therefore, the term match or matching can refer to total equivalence or similarity depending on context.

Where a reference to a device is used herein, it should be understood that the device may include or be referred to as a system, and vice versa depending on context, since a device may include a system of elements and a system may be embodied in a device.

Where a reference to a collection of elements is used herein, it should be understood that the collection of elements may include one or more elements. In some aspects or contexts, a reference to a collection of elements does not imply that the collection is an element itself.

Where a reference to an object is used herein, it should be understood that the object may be a physical object (i.e. object detected in a device's surrounding, etc.), an electronic object (i.e. object in an object oriented application program, etc.), and/or other object depending on context.

Where a mention of a function, method, routine, subroutine, or other such procedure is used herein, it should be understood that the function, method, routine, subroutine, or other such procedure comprises a call, reference, or pointer to the function, method, routine, subroutine, or other such procedure.

Where a mention of data, object, data structure, item, element, or thing is used herein, it should be understood that the data, object, data structure, item, element, or thing comprises a reference or pointer to the data, object, data structure, item, element, or thing.

Referring to Fig. 2, an embodiment of Device 98 comprising Unit for Learning and/or Using a Device's Circumstances for Autonomous Device Operation (DCADO Unit 100) is illustrated. Device 98 also comprises interconnected Processor 11, Human-machine Interface 23, Sensor 92, Object Processing Unit 93, Memory 12, and Storage 27. Processor 11 includes or executes Application Program 18. DCADO Unit 100 comprises interconnected Artificial Intelligence Unit 110, Acquisition Interface 120, and Modification Interface 130. Other additional elements can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized

in alternate embodiments.

In one example, the teaching presented by the disclosure can be implemented in a device or system for learning and/or using a device's circumstances for autonomous device operation. The device or system may include a processor circuit (i.e. Processor 11, etc.) configured to execute instruction sets (i.e. Instruction Sets 526, etc.) for operating a device. The device or system may further include a memory unit (i.e. Memory 12, etc.) configured to store data. The device or system may further include a sensor (i.e. Sensor 92, etc.) configured to detect objects (i.e. Objects 615, etc.). The device or system may further include an artificial intelligence unit (i.e. Artificial Intelligence Unit 110, etc.). The artificial intelligence unit may be configured to receive a first collection of object representations (i.e. Collection of Object Representations 525, etc.), the first collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may also be configured to receive a first one or more instruction sets for operating the device. The artificial intelligence unit may also be configured to learn the first collection of object representations correlated with the first one or more instruction sets for operating the device. The artificial intelligence unit may also be configured to receive a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may also be configured to anticipate the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may also be configured to cause the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations performed in response to the executing by the processor circuit. Any of the operations of the aforementioned elements can be performed repeatedly and/or in different orders in alternate embodiments. In some embodiments, a stream of collections of object representations can be used instead of or in addition to any collection of object representations such as, for example, using a first stream of collections of object representations instead of the first collection of object representations. In other embodiments, a logic circuit (i.e. Logic Circuit 250, etc.) may be used instead of the processor circuit. In such embodiments, one or more instruction sets for operating the device (i.e. first one or more instruction sets for operating the device, etc.) may include or be substituted with one or more inputs into or one or more outputs from the logic circuit. In further embodiments, an actuator may be included instead of or in addition to the processor circuit. In such embodiments, one or more instruction sets for operating the device (i.e. first one or more instruction sets for operating the device, etc.) may include or be substituted with one or more inputs into the actuator. Other additional elements can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate embodiments. The device or system for learning and/or using a device's circumstances for autonomous device operation may include any actions or operations of any of the disclosed methods such as methods 9100, 9200, 9300, 9400, 9500, 9600, and/or others (all later described).

Device 98 comprises any hardware, programs, or a combination thereof. Although, Device 98 is referred to as a device herein, Device 98 may be or include a system as a system may be embodied in Device 98. Device 98 may include any features, functionalities, and embodiments of Computing Device 70, or elements thereof. In some

embodiments, Device 98 includes a computing enabled device for performing mechanical or physical operations (i.e. via actuators, etc.). In other embodiments, Device 98 includes a computing enabled device for performing non-mechanical and/or other operations. Examples of Device 98 include an industrial machine, a toy, a robot, a vehicle, an appliance, a control device, a smartphone or other mobile computer, any computer, and/or other computing enabled device or machine. Such device or machine may be built for any function or purpose some examples of which are described later.

User 50 (also referred to simply as user or other suitable name or reference) comprises a human user or non-human user. A non-human User 50 includes any device, system, program, and/or other mechanism for operating or controlling Device 98 and/or elements thereof. In one example, User 50 may issue an operating direction to Application Program 18 responsive to which Application Program's 18 instructions or instruction sets may be executed by Processor 11 to perform a desired operation on Device 98. In another example, User 50 may issue an operating direction to Processor 11, Logic Circuit 250 (later described), and/or other processing element responsive to which Processor 11, Logic Circuit 250, and/or other processing element may implement logic to perform a desired operation on Device 98. User's 50 operating directions comprise any user inputted data (i.e. values, text, symbols, etc.), directions (i.e. move right, move up, move forward, copy an item, click on a link, etc.), instructions or instruction sets (i.e. manually inputted instructions or instruction sets, etc.), and/or other inputs or information. A non-human User 50 can utilize more suitable interfaces instead of, or in addition to, Human-machine Interface 23 and/or Display 21 for controlling Device 98 and/or elements thereof. Examples of such interfaces include an application programming interface (API), bridge (i.e. bridge between applications, devices, or systems, etc.), driver, socket, direct or operative connection, handle, function/routine/subroutine, and/or other interfaces.

In some embodiments, Processor 11, Logic Circuit 250, Application Program 18, and/or other processing element may control or affect an actuator (not shown). Actuator comprises the functionality for implementing motion, actions, behaviors, maneuvers, and/or other mechanical or physical operations. Device 98 may include one or more actuators to enable Device 98 to perform mechanical, physical, or other operations and/or to interact with its environment. For example, an actuator may include or be coupled to an element such as a wheel, arm, or other element to act upon the environment. Examples of an actuator include a motor, a linear motor, a servomotor, a hydraulic element, a pneumatic element, an electro-magnetic element, a spring element, and/or other actuators. Examples of types of actuators include a rotary actuator, a linear actuator, and/or other types of actuators. In other embodiments, Processor 11, Logic Circuit 250, Application Program 18, and/or other processing element may control or affect any other device or element instead of or in addition to an actuator.

Referring to Figs. 3A-3E, various embodiments of Sensors 92 and elements of Object Processing Unit 93 are illustrated.

Sensor 92 (also referred to simply as sensor or other suitable name or reference) comprises the functionality for obtaining or detecting information about its environment, and/or other functionalities. As such, one or more Sensors 92 can be used to detect objects and/or their properties in Device's 98 surrounding. In some aspects, Device's 98 surrounding may include exterior of Device 98. In other aspects, Device's 98 surrounding may include interior of Device 98 in case of hollow Device 98, Device 98 comprising compartments or openings, and/or other variously shaped Device 98. Examples of aspects of an environment that Sensor 92 can measure or be sensitive to include light (i.e. camera, lidar, etc.), electromagnetism/electromagnetic field (i.e. radar, etc.), sound (i.e.

microphone, sonar, etc.), physical contact (i.e. tactile sensor, etc.), magnetism/magnetic field (i.e. compass, etc.), electricity/electric field, temperature, gravity, vibration, pressure, and/or others. In some aspects, a passive sensor (i.e. camera, microphone, etc.) measures signals or radiation emitted or reflected by an object. In other aspects, an active sensor (i.e. lidar, radar, sonar, etc.) emits signals or radiation and measures the signals or radiation reflected or backscattered from an object. A reference to a Sensor 92 herein includes a reference to one or more Sensors 92 as applicable. In some designs, a plurality of Sensors 92 may be used to detect objects and/or their properties from different angles or sides of Device 98. For example, four Cameras 92a can be placed on four corners of Device 98 to cover 360 degrees of view of Device's 98 surrounding. In other designs, a plurality of different types of Sensors 92 may be used to detect different types of objects and/or their properties. For example, one or more Cameras 92a can be used to detect and identify an object, whereas, Radar 92d can be used to determine distance and bearing/angle of the object relative to Device 98. In further designs, a signal-emitting element can be placed within or onto an object and Sensor 92 can detect the signal from the signal-emitting element, thereby detecting the object and/or its properties. For example, a radio-frequency identification (RFID) emitter may be placed within an object to help Sensor 92 detect, identify, and/or obtain other information about the object.

In some embodiments, Sensor 92 may be or include Camera 92a as shown in Fig. 3A. Camera 92a comprises the functionality for capturing one or more pictures, and/or other functionalities. As such, Camera 92a can be used to capture pictures of Device's 98 surrounding. Camera 92a may be useful in detecting existence of an object, type of an object, identity of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In some aspects, Camera 92a may be or comprises a motion picture camera that can capture streams of pictures (i.e. motion pictures, videos, etc.). In other aspects, Camera 92a may be or comprises a still picture camera that can capture still pictures (i.e. photographs, etc.). In further aspects, Camera 92a may be or comprises a stereo camera (i.e. camera with multiple lenses, etc.) that can capture stereoscopic or range pictures. In further aspects, Camera 92a may be or comprises any other Camera 92a. In general, Camera 92a may capture any light (i.e. visible light, infrared light, ultraviolet light, x-ray light, etc.) across the electromagnetic spectrum onto a light-sensitive material. Any other technique known in art can be utilized to facilitate Camera 92a functionalities. In one example, a digital Camera 92a can utilize a charge coupled device (CCD), a complementary metal–oxide–semiconductor (CMOS) sensor, and/or other electronic image sensor to capture digital pictures that can then be stored in a memory or storage, or transmitted to any of the disclosed or other elements for further processing. In another example, analog Camera 92a can utilize an analog-to-digital converter to produce digital pictures. In some embodiments, Camera 92a can be built, embedded, or integrated in Device 98 and/or other disclosed element. In other embodiments, Camera 92a can be an external Camera 92a connected with Device 98 and/or other disclosed element. In further embodiments, Camera 92a comprises Computing Device 70 or elements thereof. In general, Camera 92a can be implemented in any suitable configuration to provide its functionalities. Camera 92a may capture one or more digital pictures. A digital picture may include a collection of color encoded pixels or dots. Examples of file formats that can be utilized to store a digital picture include JPEG, GIF, TIFF, PNG, PDF, and/or other digitally encoded picture formats. A stream of digital pictures (i.e. motion picture, video, etc.) may include one or more digital pictures. Examples of file formats that can be utilized to store a stream of digital pictures include MPEG, AVI, FLV, MOV, RM, SWF, WMV, DivX, and/or other digitally encoded motion picture formats.

In other embodiments, Sensor 92 may be or include Microphone 92b as shown in Fig. 3B. Microphone 92b comprises the functionality for capturing one or more sounds, and/or other functionalities. As such, Microphone 92b can be used to capture sounds from Device's 98 surrounding. Microphone 92b may be useful in detecting existence of an object, type of an object, identity of an object, bearing/angle of an object, activity (i.e. motion, sounding, etc.) of an object, and/or other properties of an object. In some aspects, Microphone 92b may be omnidirectional microphone that enables capturing sounds from any direction. In other aspects, Microphone 92b may be a directional (i.e. unidirectional, bidirectional, etc.) microphone that enables capturing sounds from one or more directions while ignoring or being insensitive to sounds from other directions. In general, Microphone 92b may utilize a membrane sensitive to air pressure and may produce electrical signal from air pressure variations. Samples of the electrical signal can then be read to produce a stream of digital sound samples. Any other technique known in art can be utilized to facilitate Microphone 92b functionalities. In one example, a digital Microphone 92b may include an integrated analog-to-digital converter to capture a stream of digital sound samples that can then be stored in a memory or storage, or transmitted to any of the disclosed or other elements for further processing. In another example, analog Microphone 92b may utilize an external analog-to-digital converter to produce a stream of digital sound samples. In some embodiments, Microphone 92b can be built, embedded, or integrated in Device 98. In other embodiments, Microphone 92b can be an external Microphone 92b connected with Device 98. In further embodiments where used in water, Microphone 92b may be or include a hydrophone. In further embodiments, Microphone 92b comprises Computing Device 70 or elements thereof. In general, Microphone 92b can be implemented in any suitable configuration to provide its functionalities. Examples of file formats that can be utilized to store a stream of digital sound samples include WAV, WMA, AIFF, MP3, RA, OGG, and/or other digitally encoded sound formats.

In further embodiments, Sensor 92 may be or include Lidar 92c as shown in Fig. 3C. Lidar 92c may be useful in detecting existence of an object, type of an object, identity of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In some aspects, Lidar 92c may emit a light signal (i.e. laser beam, etc.) and listen for a signal that is reflected or backscattered from an object. Any other technique known in art can be utilized to facilitate Lidar 92c functionalities.

In further embodiments, Sensor 92 may be or include a Radar 92d as shown in Fig. 3D. Radar 92d may be useful in detecting existence of an object, type of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In some aspects, Radar 92d may emit a radio signal (i.e. radio wave, etc.) and listen for a signal that is reflected or backscattered from an object. Any other technique known in art can be utilized to facilitate Radar 92d functionalities.

In further embodiments, Sensor 92 may be or include Sonar 92e as shown in Fig. 3E. Sonar 92e may be useful in detecting existence of an object, type of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In some aspects, Sonar 92e may emit a sound signal (i.e. sound pulse, etc.) and listen for a signal that is reflected or backscattered from an object. Any other technique known in art can be utilized to facilitate Sonar 92e functionalities.

One of ordinary skill in art will understand that the aforementioned sensors are described merely as examples of a variety of possible implementations, and that while all possible sensors are too voluminous to describe, other sensors known in art that can facilitate detecting of objects and/or their properties in Device's 98

surrounding are within the scope of this disclosure. Any combination of the aforementioned and/or other sensors can be used in various embodiments.

Object Processing Unit 93 comprises the functionality for processing output from Sensor 92 to obtain information of interest, and/or other functionalities. As such, Object Processing Unit 93 can be used to process output from Sensor 92 to detect objects and/or their properties in Device's 98 surrounding. In some embodiments, Object Processing Unit 93 comprises the functionality for creating or generating Collection of Object Representations 525 (also referred to as Coll of Obj Rep or other suitable name or reference) and storing one or more Object Representations 625 (also referred to simply as object representations, representations of objects, or other suitable name or reference), Object Properties 630 (also referred to simply as object properties or other suitable name or reference), and/or other elements or information into the Collection of Object Representations 525. As such, Collection of Object Representations 525 comprises the functionality for storing one or more Object Representations 625, Object Properties 630, and/or other elements or information. Object Representation 625 may include an electronic representation of an object (i.e. Object 615 [later described], etc.) detected in Device's 98 surrounding. In some aspects, Collection of Object Representations 525 includes one or more Object Representations 625, Object Properties 630, and/or other elements or information related to objects detected in Device's 98 surrounding at a particular time. Collection of Object Representations 525 may, therefore, include knowledge (i.e. unit of knowledge, etc.) of Device's 98 circumstances including objects with various properties at a particular time. In some designs, a Collection of Object Representations 525 may include or be associated with a time stamp (not shown), order (not shown), or other time related information. For example, one Collection of Object Representations 525 may be associated with time stamp t1, another Collection of Object Representations 525 may be associated with time stamp t2, and so on. Time stamps t1, t2, etc. may indicate the times of generating Collections of Object Representations 525, for instance. In other embodiments, Object Processing Unit 93 comprises the functionality for creating or generating a stream of Collections of Object Representations 525. A stream of Collections of Object Representations 525 may include one Collection of Object Representations 525 or a group, sequence, or other plurality of Collections of Object Representations 525. In some aspects, a stream of Collections of Object Representations 525 includes one or more Collections of Object Representations 525, and/or other elements or information related to objects detected in Device's 98 surrounding over time. A stream of Collections of Object Representations 525 may, therefore, include knowledge (i.e. unit of knowledge, etc.) of Device's 98 circumstances including objects with various properties over time. As circumstances including objects with various properties in Device's 98 surrounding change (i.e. objects and/or their properties change, move, act, transform, etc.) over time, this change may be captured in a stream of Collections of Object Representations 525. In some designs, each Collection of Object Representations 525 in a stream may include or be associated with the aforementioned time stamp, order, or other time related information. For example, one Collection of Object Representations 525 in a stream may be associated with order 1, a next Collection of Object Representations 525 in the stream may be associated with order 2, and so on. Orders 1, 2, etc. may indicate the orders or places of Collections of Object Representations 525 within a stream (i.e. sequence, etc.), for instance. Examples of objects include biological objects (i.e. persons, animals, vegetation, etc.), nature objects (i.e. rocks, bodies of water, etc.), manmade objects (i.e. buildings, streets, ground/aerial/aquatic vehicles, etc.), and/or others. In some aspects, any part of an object may be detected as an object itself. For instance, instead of or in addition to detecting a vehicle as

an object, a wheel and/or other parts of the vehicle may be detected as objects. In general, object may include any object or part thereof that can be detected. Examples of object properties include existence of an object, type of an object (i.e. person, cat, vehicle, building, street, tree, rock, etc.), identity of an object (i.e. name, identifier, etc.), distance of an object, bearing/angle of an object, location of an object (i.e. distance and bearing/angle from a known point, coordinates, etc.), shape/size of an object (i.e. height, width, depth, computer model, point cloud, etc.), activity of an object (i.e. motion, gestures, etc.), and/or other properties of an object. Type of an object, for example, may include any classification of objects ranging from detailed such as person, cat, vehicle, building, street, tree, rock, etc. to generalized such as biological object, nature object, manmade object, etc., and/or others including their sub-types. Location of an object, for example, can include a relative location such as one defined by distance and bearing/angle from a known point or location (i.e. Device 98 location, etc.). Location of an object, for example, can also include absolute location such as one defined by object coordinates. In general, an object property may include any attribute of an object (i.e. existence of an object, type of an object, identity of an object, shape/size of an object, etc.), any relationship of an object with Device 98, other objects, or the environment (i.e. distance of an object, bearing/angle of an object, friend/foe relationship, etc.), and/or other information related to an object. In some implementations, Object Processing Unit 93 and/or any of its elements or functionalities can be included in Sensor 92. In other implementations, Object Processing Unit 93 and/or any of its elements or functionalities can be embedded into or operate on Processor 11. In further implementations, Object Processing Unit 93 and/or any of its elements or functionalities can be embedded into or operate in DCADO Unit 100, and/or other disclosed elements. Object Processing Unit 93 may be provided in any suitable configuration. Object Processing Unit 93 may include any signal processing techniques or elements known in art as applicable.

In some embodiments, Object Processing Unit 93 may include Picture Recognizer 94a as shown in Fig. 3A. Picture Recognizer 94a comprises the functionality for detecting or recognizing objects and/or their properties in visual data, and/or other disclosed functionalities. Visual data includes digital motion pictures, digital still pictures, and/or other visual data. Examples of file formats that can be utilized to store visual data include AVI, DivX, MPEG, JPEG, GIF, TIFF, PNG, PDF, and/or other file formats. For example, Picture Recognizer 94a can be used for detecting or recognizing objects and/or their properties in one or more digital pictures captured by one or more Cameras 92a. Picture Recognizer 94a can be utilized in detecting or recognizing existence of an object, type of an object, identity of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In general, Picture Recognizer 94a can be used for any operation supported by Picture Recognizer 94a. Picture Recognizer 94a may detect or recognize an object and/or its properties as well as track the object and/or its properties in one or more digital pictures or streams of digital pictures (i.e. motion pictures, video, etc.). In the case of a person, Picture Recognizer 94a may detect or recognize a human head or face, upper body, full body, or portions/combinations thereof. In some aspects, Picture Recognizer 94a may detect or recognize objects and/or their properties from a digital picture by comparing regions of pixels from the digital picture with collections of pixels comprising known objects and/or their properties. The collections of pixels comprising known objects and/or their properties can be learned or manually, programmatically, or otherwise defined. The collections of pixels comprising known objects and/or their properties can be stored in any data structure or repository (i.e. one or more files, database, etc.) that resides locally on Device 98, or remotely on a remote computing device (i.e. server, cloud, etc.) accessible over a network or an interface. In other aspects, Picture

Recognizer 94a may detect or recognize objects and/or their properties from a digital picture by comparing features (i.e. lines, edges, ridges, corners, blobs, regions, etc.) of the digital picture with features of known objects and/or their properties. The features of known objects and/or their properties can be learned or manually, programmatically, or otherwise defined. The features of known objects and/or their properties can be stored in any data structure or repository (i.e. neural network, one or more files, database, etc.) that resides locally on Device 98, or remotely on a remote computing device (i.e. server, cloud, etc.) accessible over a network or an interface. Typical steps or elements in a feature oriented picture recognition include pre-processing, feature extraction, detection/segmentation, decision-making, and/or others, or a combination thereof, each of which may include its own sub-steps or sub-elements depending on the application. In further aspects, Picture Recognizer 94a may detect or recognize multiple objects and/or their properties from a digital picture using the aforementioned pixel or feature comparisons, and/or other detection or recognition techniques. For example, a picture may depict two objects in two of its regions both of which Picture Recognizer 94a can detect simultaneously. In further aspects, where objects and/or their properties span multiple pictures, Picture Recognizer 94a may detect or recognize objects and/or their properties by applying the aforementioned pixel or feature comparisons and/or other detection or recognition techniques over a stream of digital pictures (i.e. motion picture, video, etc.). For example, once an object is detected in a digital picture (i.e. frame, etc.) of a stream of digital pictures (i.e. motion picture, video, etc.), the region of pixels comprising the detected object or the object's features can be searched in other pictures of the stream of digital pictures, thereby tracking the object through the stream of digital pictures. In further aspects, Picture Recognizer 94a may detect or recognize an object's activities by identifying and/or analyzing differences between a detected region of pixels of one picture (i.e. frame, etc.) and detected regions of pixels of other pictures in a stream of digital pictures. For example, a region of pixels comprising a person's face can be detected in multiple consecutive pictures of a stream of digital pictures (i.e. motion picture, video, etc.). Differences among the detected regions of the consecutive pictures may be identified in the mouth part of the person's face to indicate smiling or speaking activity. In further aspects, Picture Recognizer 94a may detect or recognize objects and/or their properties using one or more artificial neural networks, which may include statistical techniques. Examples of artificial neural networks that can be used in Picture Recognizer 94a include convolutional neural networks (CNNs), time delay neural networks (TDNNs), deep neural networks, and/or others. In one example, picture recognition techniques and/or tools involving convolutional neural networks may include identifying and/or analyzing tiled and/or overlapping regions or features of a digital picture, which may then be used to search for pictures with matching regions or features. In another example, features of different convolutional neural networks responsible for spatial and temporal streams can be fused to detect objects and/or their properties in streams of digital pictures (i.e. motion pictures, videos, etc.). In general, Picture Recognizer 94a may include any machine learning, deep learning, and/or other artificial intelligence techniques. In further aspects, Picture Recognizer 94a can detect distance of a recognized object in a picture captured by a camera using structured light, sheet of light, or other lighting schemes, and/or by using phase shift analysis, time of flight, interferometry, or other techniques. In further aspects, Picture Recognizer 94a may detect distance of a recognized object in a picture captured by a stereo camera by using triangulation and/or other techniques. In further aspects, Picture Recognizer 94a may detect bearing/angle of a recognized object relative to the camera-facing direction by measuring the distance from the vertical centerline of the picture to a pixel in the recognized object based on known picture resolution and camera's angle of view. Any other techniques known in art can be utilized in Picture

Recognizer 94a. For example, thresholds for similarity, statistical techniques, and/or optimization techniques can be utilized to determine a match in any of the above-described detection or recognition techniques. In some exemplary embodiments, object recognition techniques and/or tools such as OpenCV (Open Source Computer Vision) library, CamFind API, Kooaba, 6px API, Dextro API, and/or others can be utilized for detecting or recognizing objects and/or their properties in digital pictures. In some aspects, picture recognition techniques and/or tools involve identifying and/or analyzing features such as lines, edges, ridges, corners, blobs, regions, and/or their relative positions, sizes, shapes, etc., which may then be used to search for pictures with matching features. For example, OpenCV library can detect an object (i.e. person, animal, vehicle, rock, etc.) and/or its properties in one or more digital pictures captured by Camera 92a or stored in an electronic repository, which can then be utilized in DCADO Unit 100, Artificial Intelligence Unit 110, and/or other elements. In other exemplary embodiments, facial recognition techniques and/or tools such as OpenCV (Open Source Computer Vision) library, Animetrics FaceR API, Lambda Labs Facial Recognition API, Face++ SDK, Neven Vision (also known as N-Vision) Engine, and/or others can be utilized for detecting or recognizing faces in digital pictures. In some aspects, facial recognition techniques and/or tools involve identifying and/or analyzing facial features such as the relative position, size, and/or shape of the eyes, nose, cheekbones, jaw, etc., which may then be used to search for pictures with matching features. For example, FaceR API can detect a person's face in one or more digital pictures captured by Camera 92a or stored in an electronic repository, which can then be utilized in DCADO Unit 100, Artificial Intelligence Unit 110, and/or other elements.

Various aspects or properties of digital pictures or pixels can be taken into account by Picture Recognizer 94a in any of the recognizing or comparisons. Examples of such aspects or properties include color adjustment, size adjustment, content manipulation, transparency (i.e. alpha channel, etc.), use of mask, and/or others. In some implementations, as digital pictures can be captured by various picture taking equipment, in various environments, and under various lighting conditions, Picture Recognizer 94a can adjust lighting or color of pixels or otherwise manipulate pixels before or during comparison. Lighting or color adjustment (also referred to as gray balance, neutral balance, white balance, etc.) may generally include manipulating or rebalancing the intensities of the colors (i.e. red, green, and/or blue if RGB color model is used, etc.) of one or more pixels. For example, Picture Recognizer 94a can adjust lighting or color of some or all pixels of one picture to make it more comparable to another picture. Picture Recognizer 94a can also incrementally adjust the pixels such as increasing or decreasing the red, green, and/or blue pixel values by a certain amount in each cycle of comparisons in order to find a substantially similar match at one of the incremental adjustment levels. Any of the publically available, custom, or other lighting or color adjustment techniques or programs can be utilized such as color filters, color balancing, color correction, and/or others. In other implementations, Picture Recognizer 94a can resize or otherwise transform a digital picture before or during comparison. Such resizing or transformation may include increasing or decreasing the number of pixels of a digital picture. For example, Picture Recognizer 94a can increase or decrease the size of a digital picture proportionally (i.e. increase or decrease length and/or width keeping aspect ratio constant, etc.) to equate its size with the size of another digital picture. Picture Recognizer 94a can also incrementally resize a digital picture such as increasing or decreasing the size of the digital picture proportionally by a certain amount in each cycle of comparisons in order to find a substantially similar match at one of the incremental sizes. Any of the publically available, custom, or other digital picture resizing techniques or programs can be utilized such as nearest-neighbor

interpolation, bilinear interpolation, bicubic interpolation, and/or others. In further implementations, Picture Recognizer 94a can manipulate content (i.e. all pixels, one or more regions, one or more depicted objects, etc.) of a digital picture before or during comparison. Such content manipulation may include moving, centering, aligning, resizing, transforming, and/or otherwise manipulating content of a digital picture. For example, Picture Recognizer 94a can move, center, or align content of one picture to make it more comparable to another picture. Any of the publically available, custom, or other digital picture manipulation techniques or programs can be utilized such as pixel moving, warping, distorting, aforementioned interpolations, and/or others. In further implementations, in digital pictures comprising transparency features or functionalities, Picture Recognizer 94a can utilize a threshold for acceptable number or percentage transparency difference. Alternatively, transparency can be applied to one or more pixels of a digital picture and color difference may then be determined between compared pixels taking into account the transparency related color effect. Alternatively, transparent pixels can be excluded from comparison. In further implementations, certain regions or subsets of pixels can be ignored or excluded during comparison using a mask. In general, any region or subset of a picture determined to contain no content of interest can be excluded from comparison using a mask. Examples of such regions or subsets include background, transparent or partially transparent regions, regions comprising insignificant content, or any arbitrary region or subset. Picture Recognizer 94a can perform any other pre-processing or manipulation of digital pictures or pixels before or during recognizing or comparison.

In other embodiments, Object Processing Unit 93 may include Sound Recognizer 94b as shown in Fig. 3B. Sound Recognizer 94b comprises the functionality for detecting or recognizing objects and/or their properties in audio data, and/or other disclosed functionalities. Audio data includes digital sound, and/or other audio data. Examples of file formats that can be utilized to store audio data include WAV, WMA, AIFF, MP3, RA, OGG, and/or other file formats. For example, Sound Recognizer 94b can be used for detecting or recognizing objects and/or their properties in a stream of digital sound samples captured by one or more Microphones 92b. In the case of a person, Sound Recognizer 94b may detect or recognize human voice. Sound Recognizer 94b can be utilized in detecting or recognizing existence of an object, type of an object, identity of an object, bearing/angle of an object, activity (i.e. motion, sounding, etc.) of an object, and/or other properties of an object. In general, Sound Recognizer 94b can be used for any operation supported by Sound Recognizer 94b. In some aspects, Sound Recognizer 94b may detect or recognize an object and/or its properties from a stream of digital sound samples by comparing collections of sound samples from the stream of digital sound samples with collections of sound samples of known objects and/or their properties. The collections of sound samples of known objects and/or their properties can be learned, or manually, programmatically, or otherwise defined. The collections of sound samples of known objects and/or their properties can be stored in any data structure or repository (i.e. one or more files, database, etc.) that resides locally on Device 98, or remotely on a remote computing device (i.e. server, cloud, etc.) accessible over a network or an interface. In other aspects, Sound Recognizer 94b may detect or recognize an object and/or its properties from a stream of digital sound samples by comparing features from the stream of digital sound samples with features of sounds of known objects and/or their properties. The features of sounds of known objects and/or their properties can be learned, or manually, programmatically, or otherwise defined. The features of sounds of known objects and/or their properties can be stored in any data structure or repository (i.e. one or more files, database, neural network, etc.) that resides locally on Device 98, or remotely on a remote computing device (i.e. server, cloud, etc.) accessible over

a network or an interface. Typical steps or elements in a feature oriented sound recognition include pre-processing, feature extraction, acoustic modeling, language modeling, and/or others, or a combination thereof, each of which may include its own sub-steps or sub-elements depending on the application. In further aspects, Sound Recognizer 94b may detect or recognize a variety of sounds from a stream of digital sound samples using the aforementioned sound sample or feature comparisons, and/or other detection or recognition techniques. For example, sound of a person, animal, vehicle, and/or other sounds can be detected by Sound Recognizer 94b. In further aspects, Sound Recognizer 94b may detect or recognize sounds using Hidden Markov Models (HMM), Artificial Neural Networks, Dynamic Time Warping (DTW), Gaussian Mixture Models (GMM), and/or other models or techniques, or a combination thereof. Some or all of these models or techniques may include statistical techniques. Examples of artificial neural networks that can be used in Sound Recognizer 94b include recurrent neural networks, time delay neural networks (TDNNs), deep neural networks, convolutional neural networks, and/or others. In general, Sound Recognizer 94b may include any machine learning, deep learning, and/or other artificial intelligence techniques. In further aspects, Sound Recognizer 94b may detect bearing/angle of a recognized object by measuring the direction in which Microphone 92b is pointing when sound of maximum strength is received, by analyzing amplitude of the sound, by performing phase analysis (i.e. with microphone array, etc.) of the sound, and/or by utilizing other techniques. Any other techniques known in art can be utilized in Sound Recognizer 94b. For example, thresholds for similarity, statistical techniques, and/or optimization techniques can be utilized to determine a match in any of the above-described detection or recognition techniques. In some exemplary embodiments, operating system's Sound recognition functionalities such as iOS's Voice Services, Siri, and/or others can be utilized in Sound Recognizer 94b. For example, iOS Voice Services can detect an object (i.e. person, etc.) and/or its properties in a stream of digital sound samples captured by Microphone 92b or stored in an electronic repository, which can then be utilized in DCADO Unit 100, Artificial Intelligence Unit 110, and/or other elements. In other exemplary embodiments, Java Speech API (JSAPI) implementation such as The Cloud Garden, Sphinx, and/or others can be utilized in Sound Recognizer 94b. For example, Cloud Garden JSAPI can detect an object (i.e. person, animal, vehicle, etc.) and/or its properties in a stream of digital sound samples captured by Microphone 92b or stored in an electronic repository, which can then be utilized in DCADO Unit 100, Artificial Intelligence Unit 110, and/or other elements. Any other programming language's or platform's speech or sound processing API can similarly be utilized. In further exemplary embodiments, applications or engines providing Sound recognition functionalities such as HTK (Hidden Markov Model Toolkit), Kaldi, OpenEars, Dragon Mobile, Julius, iSpeech, CeedVocal, and/or others can be utilized in Sound Recognizer 94b. For example, Kaldi SDK can detect an object (i.e. person, animal, vehicle, etc.) and/or its properties in a stream of digital sound samples captured by Microphone 92b or stored in an electronic repository, which can then be utilized in DCADO Unit 100, Artificial Intelligence Unit 110, and/or other elements.

In further embodiments, Object Processing Unit 93 may include Lidar Processing Unit 94c as shown in Fig. 3C. Lidar Processing Unit 94c comprises the functionality for detecting or recognizing objects and/or their properties using light, and/or other disclosed functionalities. As such, Lidar Processing Unit 94c can be utilized in detecting existence of an object, type of an object, identity of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In general, Lidar Processing Unit 94c can be used for any operation supported by Lidar Processing Unit 94c. In one example, Lidar Processing Unit 94c may detect distance of an object by measuring time delay between emission of a light

signal (i.e. laser beam, etc.) and return of the light signal reflected from the object based on known speed of light. In another example, Lidar Processing Unit 94c may detect bearing/angle of an object by analyzing the amplitudes of a light signal received by an array of detectors (i.e. detectors arranged into a quadrant or other arrangement, etc.). In a further example, Lidar Processing Unit 94c may detect existence, type, identity, shape/size, activity, and/or other properties of an object by illuminating the object with light and acquiring an image of the object, which can then be processed using some of the previously described or other picture recognition techniques. In a further example, Lidar Processing Unit 94c may detect existence, type, identity, shape/size, activity, and/or other properties of an object by illuminating the object with light and acquiring a point cloud representation of the object. Lidar Processing Unit 94c may detect objects and/or their properties by utilizing any lidar or light-related techniques known in art.

In further embodiments, Object Processing Unit 93 may include Radar Processing Unit 94d as shown in Fig. 3D. Radar Processing Unit 94d comprises the functionality for detecting or recognizing objects and/or their properties using radio waves, and/or other disclosed functionalities. As such, Radar Processing Unit 94d can be utilized in detecting existence of an object, type of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In general, Radar Processing Unit 94d can be used for any operation supported by Radar Processing Unit 94d. In one example, Radar Processing Unit 94d may detect existence of an object by emitting a radio signal and listening for the radio signal reflected from the object. In another example, Radar Processing Unit 94d may detect distance of an object by measuring time delay between emission of a radio signal and return of the radio signal reflected from the object based on known speed of the radio signal. In a further example, Radar Processing Unit 94d may detect bearing/angle of an object by measuring the direction in which the antenna is pointing when the return signal of maximum strength is received, by analyzing amplitude of the return signal, by performing phase analysis (i.e. with antenna array, etc.) of the return signal, and/or by utilizing any amplitude, phase, or other techniques. In a further example, Radar Processing Unit 94d may detect existence, type, identity, shape/size, activity, and/or other properties of an object by illuminating the object with radio waves and acquiring an image of the object, which can then be processed using some of the previously described or other picture recognition techniques. Radar Processing Unit 94d may detect objects and/or their properties by utilizing any radar or radio-related techniques known in art.

In further embodiments, Object Processing Unit 93 may include Sonar Processing Unit 94e as shown in Fig. 3E. Sonar Processing Unit 94e comprises the functionality for detecting or recognizing objects and/or their properties using sound, and/or other disclosed functionalities. As such, Sonar Processing Unit 94e can be utilized in detecting existence of an object, type of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In general, Sonar Processing Unit 94e can be used for any operation supported by Sonar Processing Unit 94e. In one example, Sonar Processing Unit 94e may detect existence of an object by emitting a sound signal and listening for the sound signal reflected from the object. In another example, Sonar Processing Unit 94e may detect distance of an object by measuring time delay between emission of a sound signal and return of the sound signal reflected from the object based on known speed of the sound signal. In a further example, Sonar Processing Unit 94e may detect bearing/angle of an object by measuring the direction in which the microphone is pointing when the return signal of maximum strength is received, by analyzing amplitude of the return signal, by performing phase analysis (i.e. with

microphone array, etc.) of the return signal, and/or by utilizing any amplitude, phase, or other techniques. In a further example, Sonar Processing Unit 94e may detect existence, type, identity, shape/size, activity, and/or other properties of an object by illuminating the object with sound pulses and acquiring an image of the object, which can then be processed using some of the previously described or other picture recognition techniques. Sonar Processing Unit 94e may detect objects and/or their properties by utilizing any sonar or sound-related techniques known in art.

One of ordinary skill in art will understand that the aforementioned techniques for detecting or recognizing objects and/or their properties are described merely as examples of a variety of possible implementations, and that while all possible techniques for detecting or recognizing objects and/or their properties are too voluminous to describe, other techniques for detecting or recognizing objects and/or their properties known in art are within the scope of this disclosure. Also, any signal processing technique known in art that can facilitate the disclosed functionalities can be utilized in various embodiments. Any combination of the aforementioned and/or other sensors, object detecting or recognizing techniques, signal processing techniques, and/or other elements or techniques can be used in various embodiments.

Referring to Figs. 4A-4B, an exemplary embodiment of Objects 615 (also referred to simply as objects or other suitable name or reference) detected in Device's 98 surrounding, and resulting Collection of Object Representations 525 are illustrated.

As shown for example in Fig. 4A, Object 615a is detected. Object 615a may be recognized as a cat. Object 615a may be detected at a distance of 6m from Device 98. Object 615a may be detected at a bearing/angle of 56° from Device's 98 centerline. Furthermore, Object 615b is also detected. Object 615b may be recognized as a tree. Object 615b may be detected at a distance of 10m from Device 98. Object 615b may be detected at a bearing/angle of 131° from Device's 98 centerline. Furthermore, Object 615c is also detected. Object 615c may be recognized as a person. Object 615c may be identified as John Doe. Object 615c may be detected at a distance of 8m from Device 98. Object 615c may be detected at a bearing/angle of 287° from Device's 98 centerline. Any other Objects 615 instead of or in addition to Object 615a, Object 615b, and Object 615c may be detected. In some aspects, any features, functionalities, and embodiments of Camera 92a/Picture Recognizer 94a, Microphone 92b/Sound Recognizer 94b, and/or other sensors or techniques can be utilized for recognizing and/or identifying a person, a cat, a tree, and/or other Objects 615. In further aspects, any features, functionalities, and embodiments of Camera 92a/Picture Recognizer 94a, Microphone 92b/Sound Recognizer 94b, Lidar 92c/Lidar Processing Unit 94c, Radar 92d/Radar Processing Unit 94d, Sonar 92e/Sonar Processing Unit 94e, and/or other sensors or techniques can be utilized for detecting distance, bearing/angle, and/or other object properties.

As shown for example in Fig. 4B, Object Processing Unit 93 may create or generate Collection of Object Representations 525 including Object Representation 625a representing Object 615a, Object Representation 625b representing Object 615b, Object Representation 625c representing Object 615c, etc. For instance, Object Representation 625a may include Object Property 630aa "Cat" in Category 635aa "Type", Object Property 630ab "6m" in Category 635ab "Distance", Object Property 630ac "56°" in Category 635ac "Bearing", etc. Also, Object Representation 625b may include Object Property 630ba "Tree" in Category 635ba "Type", Object Property 630bb "10m" in Category 635bb "Distance", Object Property 630bc "131°" in Category 635bc "Bearing", etc. Also, Object Representation 625c may include Object Property 630ca "Person" in Category 635ca "Type", Object Property 630cb "John Doe" in Category 635cb "Identity", Object Property 630cc "8m" in Category 635cc "Distance", Object Property

630cd “287°” in Category 635cd “Bearing”, etc. Any number of Object Representations 625, and/or other elements or information can be included in Collection of Object Representations 525. Any number of Object Properties 630 (also referred to simply as object properties or other suitable name or reference), and/or other elements or information can be included in an Object Representation 625. In some aspects, a reference to Collection of Object Representations 525 comprises a reference to a collection of Object Properties 630 and/or other elements or information related to one or more Objects 615. Other additional Object Representations 625, Object Properties 630, elements, and/or information can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate embodiments of Collection of Object Representations 525.

Referring now to DCADO Unit 100, DCADO Unit 100 comprises any hardware, programs, or a combination thereof. DCADO Unit 100 comprises the functionality for learning the operation of Device 98 in circumstances including objects with various properties. DCADO Unit 100 comprises the functionality for structuring and/or storing this knowledge in a knowledgebase (i.e. neural network, graph, sequences, other repository, etc.). DCADO Unit 100 comprises the functionality for enabling autonomous operation of Device 98 in circumstances including objects with various properties. DCADO Unit 100 comprises the functionality for interfacing with or attaching to Application Program 18, Processor 11, Logic Circuit 250 (later described), and/or other processing element. DCADO Unit 100 comprises the functionality for obtaining instruction sets, data, and/or other information used, implemented, and/or executed by Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. DCADO Unit 100 comprises the functionality for modifying instruction sets, data, and/or other information used, implemented, and/or executed by Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. DCADO Unit 100 comprises learning, anticipating, decision making, automation, and/or other functionalities disclosed herein. Statistical, artificial intelligence, machine learning, and/or other models or techniques are utilized to implement the disclosed devices, systems, and methods.

When DCADO Unit 100 functionalities are applied on Application Program 18, Processor 11, Logic Circuit 250 (later described), and/or other processing element of Device 98, Device 98 may become autonomous. DCADO Unit 100 may take control from, share control with, and/or release control to Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element to implement autonomous operation of Device 98. DCADO Unit 100 may take control from, share control with, and/or release control to Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element automatically or after prompting User 50 to allow it. In some aspects, Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element of an autonomous Device 98 may include or be provided with anticipatory (also referred to as alternate or other suitable name or reference) instructions or instruction sets that User 50 did not issue or cause to be executed. Such anticipatory instructions or instruction sets include instruction sets that User 50 may want or is likely to issue or cause to be executed. Anticipatory instructions or instruction sets can be generated by DCADO Unit 100 or elements thereof based on Device's 98 circumstances including objects with various properties. As such, Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element of an autonomous Device 98 may include or be provided with some or all original instructions or instruction sets and/or any anticipatory instructions or instruction sets generated by DCADO Unit 100. Therefore, autonomous Device 98 operating may include executing some or all original instructions or instruction sets and/or any anticipatory instructions or instruction sets generated by DCADO Unit 100. In one example, DCADO Unit 100 can overwrite or rewrite the original instructions or instruction sets of Application

Program 18, Processor 11, Logic Circuit 250, and/or other processing element with DCADO Unit 100-generated instructions or instruction sets. In another example, DCADO Unit 100 can insert or embed DCADO Unit 100-generated instructions or instruction sets among the original instructions or instruction sets of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. In a further example, DCADO Unit 100 can branch, redirect, or jump to DCADO Unit 100-generated instructions or instruction sets from the original instructions or instruction sets of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element.

In some embodiments, autonomous Device 98 operating comprises determining, by DCADO Unit 100, a next instruction or instruction set to be executed based on Device's 98 circumstances including objects with various properties prior to the user issuing or causing to be executed the next instruction or instruction set. In yet other embodiments, autonomous application operating comprises determining, by DCADO Unit 100, a next instruction or instruction set to be executed based on Device's 98 circumstances including objects with various properties prior to the system receiving the next instruction or instruction set.

In some embodiments, autonomous Device 98 operating includes a partially or fully autonomous operating. In an example involving partially autonomous Device 98 operating, a user confirms DCADO Unit 100-generated instructions or instruction sets prior to their execution. In an example involving fully autonomous application operating, DCADO Unit 100-generated instructions or instruction sets are executed without user or other system confirmation (i.e. automatically, etc.).

In some embodiments, a combination of DCADO Unit 100 and other systems and/or techniques can be utilized to implement Device's 98 operation. In one example, DCADO Unit 100 may be a primary or preferred system for implementing Device's 98 operation. While operating autonomously under the control of DCADO Unit 100, Device 98 may encounter a circumstance including objects with various properties that has not been encountered or learned before. In such situations, User 50 and/or non-DCADO system may take control of Device's 98 operation. DCADO Unit 100 may take control again when Device 98 encounters a previously learned circumstance including objects with various properties. Naturally, DCADO Unit 100 can learn Device's 98 operation in circumstances while User 50 and/or non-DCADO system is in control of Device 98, thereby reducing or eliminating the need for future involvement of User 50 and/or non-DCADO system. In another example, User 50 and/or non-DCADO system may be a primary or preferred system for implementing Device's 98 operation. While operating under the control of User 50 and/or non-DCADO system, User 50 and/or non-DCADO system may release control to DCADO Unit 100 for any reason (i.e. User 50 gets tired or distracted, non-DCADO system gets stuck or cannot make a decision, etc.), at which point Device 98 can be controlled by DCADO Unit 100. In some designs, DCADO Unit 100 may take control in certain special circumstances including objects with various properties where DCADO Unit 100 may offer superior performance even though User 50 and/or non-DCADO system may generally be preferred. Once Device 98 leaves such special circumstances, DCADO Unit 100 may release control to User 50 and/or non-DCADO system. In general, DCADO Unit 100 can take control from, share control with, or release control to User 50, non-DCADO system, and/or other system or process at any time, in any circumstances, and remain in control for any period of time as needed.

In some embodiments, DCADO Unit 100 may control one or more sub-devices, sub-systems, or elements of Device 98 while User 50 and/or non-DCADO system may control other one or more sub-devices, sub-systems, or elements of Device 98.

It should be understood that a reference to autonomous operating of Device 98 may include autonomous operating of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element depending on context.

Referring now to Acquisition Interface 120, Acquisition Interface 120 comprises the functionality for obtaining and/or receiving instruction sets, data, and/or other information. Acquisition Interface 120 comprises the functionality for obtaining and/or receiving instruction sets, data, and/or other information from Processor 11, Application Program 18, Logic Circuit 250 (later described), and/or other processing element. Acquisition Interface 120 comprises the functionality for obtaining and/or receiving instruction sets, data, and/or other information at runtime. In some aspects, an instruction set may include any computer command, instruction, signal, or input used in Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. Therefore, the terms instruction set, command, instruction, signal, input, or other such terms may be used interchangeably herein depending on context. Acquisition Interface 120 also comprises the functionality for attaching to or interfacing with Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. In one example, Acquisition Interface 120 comprises the functionality to access and/or read runtime engine/environment, virtual machine, operating system, compiler, just-in-time (JIT) compiler, interpreter, translator, execution stack, file, object, data structure, and/or other computing system elements. In another example, Acquisition Interface 120 comprises the functionality to access and/or read memory, storage, bus, interfaces, and/or other computing system elements. In a further example, Acquisition Interface 120 comprises the functionality to access and/or read Processor 11 registers and/or other Processor 11 elements. In a further example, Acquisition Interface 120 comprises the functionality to access and/or read inputs and/or outputs of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. In a further example, Acquisition Interface 120 comprises the functionality to access and/or read functions, methods, procedures, routines, subroutines, and/or other elements of Application Program 18. In a further example, Acquisition Interface 120 comprises the functionality to access and/or read source code, bytecode, compiled, interpreted, or otherwise translated code, machine code, and/or other code. In a further example, Acquisition Interface 120 comprises the functionality to access and/or read values, variables, parameters, and/or other data or information. Acquisition Interface 120 also comprises the functionality for transmitting the obtained instruction sets, data, and/or other information to Artificial Intelligence Unit 110 and/or other element. As such, Acquisition Interface 120 provides input into Artificial Intelligence Unit 110 for knowledge structuring, anticipating, decision making, and/or other functionalities later in the process. Acquisition Interface 120 also comprises other disclosed functionalities.

Acquisition Interface 120 can employ various techniques for obtaining instruction sets, data, and/or other information. In one example, Acquisition Interface 120 can attach to and/or obtain Processor's 11, Application Program's 18, Logic Circuit's 250, and/or other processing element's instruction sets, data, and/or other information through tracing or profiling techniques. Tracing or profiling may be used for outputting Processor's 11, Application Program's 18, Logic Circuit's 250, and/or other processing element's instruction sets, data, and/or other information at runtime. For instance, tracing or profiling may include adding trace code (i.e. instrumentation, etc.) to an application and/or outputting trace information to a specific target. The outputted trace information (i.e. instruction sets, data, and/or other information, etc.) can then be provided to or recorded into a file, data structure, repository, an application, and/or other system or target that may receive such trace information. As such, Acquisition Interface

120 can utilize tracing or profiling to obtain instruction sets, data, and/or other information and provide them as input into Artificial Intelligence Unit 110. In some aspects, instrumentation can be performed in source code, bytecode, compiled, interpreted, or otherwise translated code, machine code, and/or other code. In other aspects, instrumentation can be performed in various elements of a computing system such as memory, virtual machine, runtime engine/environment, operating system, compiler, interpreter, translator, processor registers, execution stack, program counter, and/or other elements. In yet other aspects, instrumentation can be performed in various abstraction layers of a computing system such as in software layer (i.e. Application Program 18, etc.), in virtual machine (if VM is used), in operating system, in Processor 11, and/or in other layers or areas that may exist in a particular computing system implementation. In yet other aspects, instrumentation can be performed at various time periods in an application's execution such as source code write time, compile time, interpretation time, translation time, linking time, loading time, runtime, and/or other time periods. In yet other aspects, instrumentation can be performed at various granularities or code segments such as some or all lines of code, some or all statements, some or all instructions or instruction sets, some or all basic blocks, some or all functions/routines/subroutines, and/or some or all other code segments.

In some embodiments, Application Program 18 can be automatically instrumented. For example, Acquisition Interface 120 can access Application Program's 18 source code, bytecode, or machine code and select instrumentation points of interest. Selecting instrumentation points may include finding locations in the source code, bytecode, or machine code corresponding to function calls, function entries, function exits, object creations, object destructions, event handler calls, new lines (i.e. to instrument all lines of code, etc.), thread creations, throws, and/or other points of interest. Instrumentation code can then be inserted at the instrumentation points of interest to output Application Program's 18 instruction sets, data, and/or other information. In response to executing instrumentation code, Application Program's 18 instruction sets, data, and/or other information may be received by Acquisition Interface 120. In some aspects, Application Program's 18 source code, bytecode, or machine code can be dynamically instrumented. For example, instrumentation code can be dynamically inserted into Application Program 18 at runtime.

In other embodiments, Application Program 18 can be manually instrumented. In one example, a programmer can instrument a function call by placing an instrumenting instruction immediately after the function call as in the following example.

```
Device1.moveForward(12);
traceApplication('Device1.moveForward(12);');
```

In another example, an instrumenting instruction can be placed immediately before the function call, or at the beginning, end, or anywhere within the function itself. A programmer may instrument all function calls or only function calls of interest. In a further example, a programmer can instrument all lines of code or only code lines of interest. In a further example, a programmer can instrument other elements utilized or implemented within Application Program 18 such as objects and/or any of their functions, data structures and/or any of their functions, event handlers and/or any of their functions, threads and/or any of their functions, and/or other elements or functions. Similar instrumentation as in the preceding examples can be performed automatically or dynamically. In some designs where manual code instrumentation is utilized, Acquisition Interface 120 can optionally be omitted and Application Program's 18 instruction sets, data, and/or other information may be transmitted directly to Artificial

Intelligence Unit 110.

In some embodiments, DCADO Unit 100 can be selective in learning instruction sets, data, and/or other information to those implemented, utilized, or related to an object, data structure, repository, thread, function, and/or other element of Application Program 18. In some aspects, Acquisition Interface 120 can obtain Application Program's 18 instruction sets, data, and/or other information implemented, utilized, or related to a certain object in an object oriented Application Program 18.

In some embodiments, various computing systems and/or platforms may provide native tools for obtaining instruction sets, data, and/or other information. Also, independent vendors may provide portable tools with similar functionalities that can be utilized across different computing systems and/or platforms. These native and portable tools may provide a wide range of functionalities to obtain runtime and other information such as instrumentation, tracing or profiling, logging application or system messages, outputting custom text messages, outputting objects or data structures, outputting functions/routines/subroutines or their invocations, outputting variable or parameter values, outputting thread or process behaviors, outputting call or other stacks, outputting processor registers, providing runtime memory access, providing inputs and/or outputs, performing live application monitoring, and/or other capabilities. One of ordinary skill in art will understand that, while all possible variations of the techniques to obtain instruction sets, data, and/or other information are too voluminous to describe, these techniques are within the scope of this disclosure.

In one example, obtaining instruction sets, data, and/or other information can be implemented through the .NET platform's native tools for application tracing or profiling such as System.Diagnostics.Trace, System.Diagnostics.Debug, and System.Diagnostics.TraceSource classes for tracing execution flow, and System.Diagnostics.Process, System.Diagnostics.EventLog, and System.Diagnostics.PerformanceCounter classes for profiling code, accessing local and remote processes, starting and stopping system processes, and interacting with Windows event logs, etc. For instance, a set of trace switches can be created that output an application's information. The switches can be configured using the .config file. For a Web application, this may typically be Web.config file associated with the project. In a Windows application, this file may typically be named applicationName.exe.config. Trace code can be added to application code automatically or manually as previously described. Appropriate listener can be created where the trace output is received. Trace code may output trace messages to a specific target such as a file, a log, a database, an object, a data structure, and/or other repository or system. Acquisition Interface 120 or Artificial Intelligence Unit 110 can then read or obtain the trace information from these targets. In some aspects, trace code may output trace messages directly to Acquisition Interface 120. In other aspects, trace code may output trace messages directly to Artificial Intelligence Unit 110. In the case of outputting trace messages to Acquisition Interface 120 or directly to Artificial Intelligence Unit 110, custom listeners can be built to accommodate these specific targets. Other platforms, tools, and/or techniques can provide equivalent or similar functionalities as the above described ones.

In another example, obtaining instruction sets, data, and/or other information can be implemented through the .NET platform's Profiling API that can be used to create a custom profiler application for tracing, monitoring, interfacing with, and/or managing a profiled application. The Profiling API provides an interface that includes methods to notify the profiler of events in the profiled application. The Profiling API may also provide an interface to enable the profiler to call back into the profiled application to obtain information about the state of the profiled

application. The Profiling API may further provide call stack profiling functionalities. Call stack (also referred to as execution stack, control stack, runtime stack, machine stack, the stack, etc.) includes a data structure that can store information about active subroutines of an application. The Profiling API may provide a stack snapshot method, which enables a trace of the stack at a particular point in time. The Profiling API may also provide a shadow stack method, which tracks the call stack at every instant. A shadow stack can obtain function arguments, return values, and information about generic instantiations. A function such as FunctionEnter can be utilized to notify the profiler that control is being passed to a function and can provide information about the stack frame and function arguments. A function such as FunctionLeave can be utilized to notify the profiler that a function is about to return to the caller and can provide information about the stack frame and function return value. An alternative to call stack profiling includes call stack sampling in which the profiler can periodically examine the stack. In some aspects, the Profiling API enables the profiler to change the in-memory code stream for a routine before it is just-in-time (JIT) compiled where the profiler can dynamically add instrumentation code to all or particular routines of interest. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through Java platform's APIs for application tracing or profiling such as Java Virtual Machine Profiling Interface (JVMPI), Java Virtual Machine Tool Interface (JVMTI), and/or other APIs or tools. These APIs can be used for instrumentation of an application, for notification of Java Virtual Machine (VM) events, and/or other functionalities. One of the tracing or profiling techniques that can be utilized includes bytecode instrumentation. The profiler can insert bytecodes into all or some of the classes. In application execution profiling, for example, these bytecodes may include methodEntry and methodExit calls. In memory profiling, for example, the bytecodes may be inserted after each new or after each constructor. In some aspects, insertion of instrumentation bytecode can be performed either by a post-compiler or a custom class loader. An alternative to bytecode instrumentation includes monitoring events generated by the JVMPI or JVMTI interfaces. Both APIs can generate events for method entry/exit, object allocation, and/or other events. In some aspects, JVMTI can be utilized for dynamic bytecode instrumentation where insertion of instrumentation bytecodes is performed at runtime. The profiler may insert the necessary instrumentation when a selected class is invoked in an application. This can be accomplished using the JVMTI's redefineClasses method, for example. This approach also enables changing of the level of profiling as the application is running. If needed, these changes can be made adaptively without restarting the application. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through JVMTI's programming interface that enables creation of software agents that can monitor and control a Java application. An agent may use the functionality of the interface to register for notification of events as they occur in the application, and to query and control the application. A JVMTI agent may use JVMTI functions to extract information from a Java application. A JVMTI agent can be utilized to obtain an application's runtime information such as method calls, memory allocation, CPU utilization, lock contention, and/or other information. JVMTI may include functions to obtain information about variables, fields, methods, classes, and/or other information. JVMTI may also provide notification for numerous events such as method entry and exit, exception, field access and modification, thread start and end, and/or other events. Examples of JVMTI built-in methods include GetMethodName to obtain the name of an invoked method, GetThreadInfo to obtain information for a specific

thread, GetClassSignature to obtain information about the class of an object, GetStackTrace to obtain information about the stack including information about stack frames, and/or other methods. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through java.lang.Runtime class that provides an interface for application tracing or profiling. Examples of methods provided in java.lang.Runtime that can be used to obtain an application's instruction sets, data, and/or other information include tracemethodcalls, traceinstructions, and/or other methods. These methods prompt the Java Virtual Machine to output trace information for a method or instruction in the virtual machine as it is executed. The destination of trace output may be system dependent and include a file, a listener, and/or other destinations where Acquisition Interface 120, Artificial Intelligence Unit 110, and/or other disclosed elements can access needed information. In addition to tracing or profiling tools native to their respective computing systems and/or platforms, many independent tools exist that provide tracing or profiling functionalities on more than one computing system and/or platform. Examples of these tools include Pin, DynamoRIO, KernInst, DynInst, Kprobes, OpenPAT, DTrace, SystemTap, and/or others. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through logging tools of the platform and/or operating system on which an application runs. Some logging tools may include nearly full feature sets of the tracing or profiling tools previously described. In one example, Visual Basic enables logging of runtime messages through its Microsoft.VisualBasic.Logging namespace that provides a log listener where the log listener may direct logging output to a file and/or other target. In another example, Java enables logging through its java.util.logging class. In some aspects, obtaining an application's instruction sets, data, and/or other information can be implemented through logging capabilities of the operating system on which an application runs. For example, Windows NT features centralized log service that applications and operating-system components can utilize to report their events including any messages. Windows NT provides functionalities for system, application, security, and/or other logging. An application log may include events logged by applications. Windows NT, for example, may include support for defining an event source (i.e. application that created the event, etc.). Windows Vista, for example, supports a structured XML log-format and designated log types to allow applications to more precisely log events and to help interpret the events. Examples of different types of event logs include administrative, operational, analytic, debug, and/or other log types including any of their subcategories. Examples of event attributes that can be utilized include eventID, level, task, opcode, keywords, and/or other event attributes. Windows wevtutil tool enables access to events, their structures, registered event publishers, and/or their configuration even before the events are fired. Wevtutil supports capabilities such as retrieval of the names of all logs on a computing device; retrieval of configuration information for a specific log; retrieval of event publishers on a computing device; reading events from an event log, from a log file, or using a structured query; exporting events from an event log, from a log file, or using a structured query to a specific target; and/or other capabilities. Operating system logs can be utilized solely if they contain sufficient information on an application's instruction sets, data, and/or other information. Alternatively, operating system logs can be utilized in combination with another source of information (i.e. trace information, call stack, processor registers, memory, etc.) to reconstruct the application's instruction sets, data, and/or other information needed for Artificial Intelligence Unit 110 and/or other elements. In

addition to logging capabilities native to their respective platforms and/or operating systems, many independent tools exist that provide logging on different platforms and/or operating systems. Examples of these tools include Log4j, Logback, SmartInspect, NLog, log4net, Microsoft Enterprise Library, ObjectGuy Framework, and/or others. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through tracing or profiling the operating system on which an application runs. As in tracing or profiling an application, one of the techniques that can be utilized includes adding instrumentation code to the operating system's source code. Such instrumentation code can be added to the operating system's source code before kernel compilation or recompilation, for instance. This type of instrumentation may involve defining or finding locations in the operating system's source code where instrumentation code may be inserted. Kernel instrumentation can also be performed without the need for kernel recompilation or rebooting. In some aspects, instrumentation code can be added at locations of interest through binary rewriting of compiled kernel code. In other aspects, kernel instrumentation can be performed dynamically where instrumentation code is added and/or removed where needed at runtime. For instance, dynamic instrumentation may overwrite kernel code with a branch instruction that redirects execution to instrumentation code or instrumentation routine. In yet other aspects, kernel instrumentation can be performed using just-in-time (JIT) dynamic instrumentation where execution may be redirected to a copy of kernel's code segment that includes instrumentation code. This type of instrumentation may include a JIT compiler and creation of a copy of the original code segment having instrumentation code or calls to instrumentation routines embedded into the original code segment. Instrumentation of the operating system may enable total system visibility including visibility into an application's behavior by enabling generation of low level trace information. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through tracing or profiling the processor on which an application runs. For example, some Intel processors provide Intel Processor Trace (i.e. Intel PT, etc.), a low-level tracing feature that enables recording executed instruction sets, and/or other data or information of one or more applications. Intel PT is facilitated by the Processor Trace Decoder Library along with its related tools. Intel PT is a low-overhead execution tracing feature that records information about application execution on each hardware thread using dedicated hardware facilities. The recorded execution/trace information is collected in data packets that can be buffered internally before being sent to a memory subsystem or another system or element (i.e. Acquisition Interface 120, Artificial Intelligence Unit 110, etc.). Intel PT also enables navigating the recorded execution/trace information via reverse stepping commands. Intel PT can be included in an operating system's core files and provided as a feature of the operating system. Intel PT can trace globally some or all applications running on an operating system. Acquisition Interface 120 or Artificial Intelligence Unit 110 can read or obtain the recorded execution/trace information from Intel PT. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through branch tracing or profiling. Branch tracing may include an abbreviated instruction trace in which only the successful branch instruction sets are traced or recorded. Branch tracing can be implemented through utilizing dedicated processor commands, for example. Executed branches may be saved into special branch trace store area of

memory. With the availability and reference to a compiler listing of the application together with branch trace information, a full path of executed instruction sets can be reconstructed. The full path can also be reconstructed with a memory dump (containing the program storage) and branch trace information. In some aspects, branch tracing can be utilized for pre-learning or automated learning of an application's instruction sets, data, and/or other information where a number of application simulations (i.e. simulations of likely/common operations, etc.) are performed. As such, the application's operation can be learned automatically saving the time that would be needed to learn the application's operation directed by a user. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through assembly language. Assembly language is a low-level programming language for a computer or other programmable device in which there is a strong correlation between the language and the architecture's machine instruction sets. Syntax, addressing modes, operands, and/or other elements of an assembly language instruction set may translate directly into numeric (i.e. binary, etc.) representations of that particular instruction set. Because of this direct relationship with the architecture's machine instruction sets, assembly language can be a powerful tool for tracing or profiling an application's execution in processor registers, memory, and/or other computing system components. For example, using assembly language, memory locations of a loaded application can be accessed, instrumented, and/or otherwise manipulated. In some aspects, assembly language can be used to rewrite or overwrite original in-memory instruction sets of an application with instrumentation instruction sets. In other aspects, assembly language can be used to redirect application's execution to instrumentation routine/subroutine or other code segment elsewhere in memory by inserting a jump into the application's in-memory code, by redirecting program counter, or by other techniques. Some operating systems may implement protection from changes to applications loaded into memory. Operating system, processor, or other low level commands such as Linux mprotect command or similar commands in other operating systems may be used to unprotect the protected locations in memory before the change. In yet other aspects, assembly language can be used to obtain instruction sets, data, and/or other information through accessing and/or reading instruction register, program counter, other processor registers, memory locations, and/or other components of a computing system. In yet other aspects, high-level programming languages may call or execute an external assembly language program to facilitate obtaining instruction sets, data, and/or other information as previously described. In yet other aspects, relatively low-level programming languages such as C may allow embedding assembly language directly in their source code such as, for example, using asm keyword of C. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, it may be sufficient to obtain user or other inputs, variables, parameters, and/or other data in some procedural, simple object oriented, or other applications. In one instance, a simple procedural application executes a sequence of instruction sets until the end of the program. During its execution, the application may receive user or other input, store the input in a variable, and perform calculations using the variable to reach a result. The value of the variable can be obtained or traced. In another instance, a more complex procedural application comprises one or more functions/routines/subroutines each of which may include a sequence of instruction sets. The application may execute a main sequence of instruction sets with a branch to a function/routine/subroutine. During its execution, the application may receive user or other input, store the input in a

variable, and pass the variable as a parameter to the function/routine/subroutine. The function/routine/subroutine may perform calculations using the parameter and return a value that the rest of the application can use to reach a result. The value of the variable or parameter passed to the function/routine/subroutine, and/or return value can be obtained or traced. Values of user or other inputs, variables, parameters, and/or other items of interest can be obtained through previously described tracing, instrumentation, and/or other techniques. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

Referring to Fig. 5, in yet another example, obtaining instruction sets, data, and/or other information may be implemented through tracing, profiling, or sampling of instruction sets or data in processor registers, memory, or other computing system components where instruction sets, data, and/or other information may be stored or utilized. For example, Instruction Register 212 may be part of Processor 11 and it may store the instruction set currently being executed or decoded. In some processors, Program Counter 211 (also referred to as instruction pointer, instruction address register, instruction counter, or part of instruction sequencer) may be incremented after fetching an instruction set, and it may hold or point to the memory address of the next instruction set to be executed. In a processor where the incrementation precedes the fetch, Program Counter 211 may point to the current instruction set being executed. In the instruction cycle, an instruction set may be loaded into Instruction Register 212 after Processor 11 fetches it from location in Memory 12 pointed to by Program Counter 211. Instruction Register 212 may hold the instruction set while it is decoded by Instruction Decoder 213, prepared, and executed. In some aspects, data (i.e. operands, etc.) needed for instruction set execution may be loaded from Memory 12 into a register within Register Array 214. In other aspects, the data may be loaded directly into Arithmetic Logic Unit 215. For instance, as instruction sets pass through Instruction Register 212 during application execution, they may be transmitted to Acquisition Interface 120 as shown. Examples of the steps in execution of a machine instruction set may include decoding the opcode (i.e. portion of a machine instruction set that may specify the operation to be performed), determining where the operands may be located (depending on architecture, operands may be in registers, the stack, memory, I/O ports, etc.), retrieving the operands, allocating processor resources to execute the instruction set (needed in some types of processors), performing the operation indicated by the instruction set, saving the results of execution, and/or other execution steps. Examples of the types of machine instruction sets that can be utilized include arithmetic, data handling, logical, program control, as well as special and/or other instruction set types. In addition to the ones described or shown, examples of other computing system or processor components that can be used during an instruction cycle include memory address register (MAR) that may hold the address of a memory block to be read from or written to; memory data register (MDR) that may hold data fetched from memory or data waiting to be stored in memory; data registers that may hold numeric values, characters, small bit arrays, or other data; address registers that may hold addresses used by instruction sets that indirectly access memory; general purpose registers (GPRs) that may store both data and addresses; conditional registers that may hold truth values often used to determine whether some instruction set should or should not be executed; floating point registers (FPRs) that may store floating point numbers; constant registers that may hold read-only values such as zero, one, or pi; special purpose registers (SPRs) such as status register, program counter, or stack pointer that may hold information on program state; machine-specific registers that may store data and settings related to a particular processor; Register Array 214 that may include an array of any number of processor registers; Arithmetic Logic Unit 215 that may perform arithmetic and logic operations; control unit that may direct processor's operation;

and/or other circuits or components. Tracing, profiling, or sampling of processor registers, memory, or other computing system components can be implemented in a program, combination of hardware and program, or purely hardware system. Dedicated hardware may be built to perform tracing, profiling, or sampling of processor registers or any computing system components with marginal or no impact to computing overhead.

One of ordinary skill in art will recognize that Fig. 5 depicts one of many implementations of processor or computing system components, and that various additional components can be included, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate implementations. Processor or computing system components may be arranged or connected differently in alternate implementations. Processor or computing system components may also be connected with external elements using various connections. For instance, the connection between Instruction Register 212 and Acquisition Interface 120 may include any number or types of connections such as, for example, a dedicated connection for each bit of Instruction Register 212 (i.e. 32 connections for a 32 bit Instruction Register 212, etc.). Any of the described or other connections or interfaces may be implemented among any processor or computing system components and Acquisition Interface 120 or other elements.

Referring to Figs. 6A-6B, in yet another example, obtaining instruction sets, data, and/or other information may be implemented through tracing, profiling, or sampling of Logic Circuit 250. While Processor 11 includes any type or embodiment of logic circuit, Logic Circuit 250 is described separately here to offer additional detail on its functioning. Some Devices 98 may not need the processing capabilities of an entire Processor 11, but instead a more tailored Logic Circuit 250. Examples of such Devices 98 include home appliances, audio or video electronics, vehicle systems, toys, industrial machines, robots, and/or others. Logic Circuit 250 comprises the functionality for performing logic operations. Logic Circuit 250 comprises the functionality for performing logic operations using the circuit's inputs and producing outputs based on the logic operations performed on the inputs. Logic Circuit 250 may generally be implemented using transistors, diodes, and/or other electronic switches, but can also be constructed using vacuum tubes, electromagnetic relays (relay logic), fluidic logic, pneumatic logic, optics, molecules, or even mechanical elements. In some aspects, Logic Circuit 250 may be or include a microcontroller, field-programmable gate array (FPGA), application-specific integrated circuit (ASIC), and/or other computing circuit or device. In other aspects, Logic Circuit 250 may be or include any circuit or device comprising one or more logic gates, one or more transistors, one or more switches, and/or one or more other logic components. In further aspects, Logic Circuit 250 may be or include any integrated or other circuit or device that can perform logic operations. Logic may generally refer to Boolean logic utilized in binary operations, but other logics can also be used. Input into Logic Circuit 250 may include or refer to a value inputted into the Logic Circuit 250, therefore, these terms may be used interchangeably herein depending on context. In one example, Logic Circuit 250 may perform some logic operations using four input values and produce two output values. As the four input values are delivered to or received by Logic Circuit 250, they may be obtained by Acquisition Interface 120 through the four hardwired connections as shown in Fig. 6A. In another example, Logic Circuit 250 may perform some logic operations using four input values and produce two output values. As the two output values are generated by or transmitted out of Logic Circuit 250, they may be obtained by Acquisition Interface 120 through the two hardwired connections as shown in Fig. 6B. In a further example, instead of or in addition to obtaining input and/or output values of Logic Circuit 250, the state of Logic Circuit 250 may be obtained by reading or accessing values from one or more Logic Circuit's 250 internal

components such as registers, memories, buses, and/or others (i.e. similar to the previously described tracing, profiling, and/or sampling of Processor 11 components, etc.). Tracing, profiling, or sampling of Logic Circuit 250 can be implemented in a program, combination of hardware and program, or purely hardware system. Dedicated hardware may be built to perform tracing, profiling, or sampling of Logic Circuit 250 with marginal or no impact to computing overhead. Any of the elements and/or techniques for tracing, profiling, or sampling of Logic Circuit 250 can similarly be implemented with Processor 11 and/or other processing elements. In some designs, DCADO Unit 100 may include clamps and/or other elements to attach DCADO Unit 100 to inputs (i.e. input wires, etc.) into and/or outputs (i.e. output wires, etc.) from Logic Circuit 250. Such clamps and/or attachment elements enable seamless attachment of DCADO Unit 100 to any circuit or computing device without the need to redesign or alter the circuit or computing device.

In some embodiments, DCADO Unit 100 may learn input values directly from an actuator (previously described, not shown). For example, Logic Circuit 250 or other processing element may control an actuator that enables Device 98 to perform mechanical, physical, and/or other operations. An actuator may receive one or more input values or control signals from Logic Circuit 250 or other processing element directing the actuator to perform specific operations. As one or more input values or control signals are delivered to or received by the actuator, they may be obtained by Acquisition Interface 120 as previously described with respect to obtaining input values of Logic Circuit 250. Specifically, for instance, one or more input values or control signals of an actuator may be obtained by Acquisition Interface 120 via hardwired or other connections.

One of ordinary skill in art will recognize that Figs. 6A-6B depict one of many implementations of Logic Circuit 250 and that any number of input and/or output values can be utilized in alternate implementations. One of ordinary skill in art will also recognize that Logic Circuit 250 may include any number and/or combination of logic components to implement any logic operations.

Other additional techniques or elements may be utilized as needed for obtaining instruction sets, data, and/or other information, or some of the disclosed techniques or elements may be excluded, or a combination thereof may be utilized in alternate embodiments.

Referring to Figs. 7A-7E, some embodiments of Instruction Sets 526 are illustrated. In some aspects, Instruction Set 526 includes one or more instructions or commands of Application Program 18. For example, Instruction Set 526 may include one or more instructions or commands of a high-level programming language such as Java or SQL, a low-level language such as assembly or machine language, an intermediate language or construct such as bytecode, and/or any other language or construct. In other aspects, Instruction Set 526 includes one or more inputs into and/or outputs from Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element. In further aspects, Instruction Set 526 includes one or more values or states of registers and/or other components of Logic Circuit 250, Processor 11, and/or other processing element. In general, Instruction Set 526 may include one or more instructions, commands, keywords, symbols (i.e. parentheses, brackets, commas, semicolons, etc.), operators (i.e. =, <, >, etc.), variables, values, objects, data structures, functions (i.e. Function1(), FIRST(), MIN(), SQRT(), etc.), parameters, states, signals, inputs, outputs, characters, digits, references thereto, and/or other components for performing an operation.

In an embodiment shown in Fig. 7A, Instruction Set 526 includes code of a high-level programming language (i.e. Java, C++, etc.) comprising the following function call construct: Function1 (Parameter1, Parameter2,

Parameter3, ...). An example of a function call applying the above construct includes the following Instruction Set 526: moveTo(Device1, 14, 8). The function or reference thereto "moveTo(Device1, 14, 8)" may be an Instruction Set 526 directing Device1 to move to a location with coordinates 14 and 8, for example. In another embodiment shown in Fig. 7B, Instruction Set 526 includes structured query language (SQL). In a further embodiment shown in Fig. 7C, Instruction Set 526 includes bytecode (i.e. Java bytecode, Python bytecode, CLR bytecode, etc.). In a further embodiment shown in Fig. 7D, Instruction Set 526 includes assembly code. In a further embodiment shown in Fig. 7E, Instruction Set 526 includes machine code.

Referring to Figs. 8A-8B, some embodiments of Extra Information 527 (also referred to as Extra Info 527) are illustrated. In an embodiment shown in Fig. 8A, Collection of Object Representations 525 may include or be associated with Extra Info 527. In an embodiment shown in Fig. 8B, Instruction Set 526 may include or be associated with Extra Info 527.

Extra Info 527 comprises the functionality for storing any information useful in comparisons or decision making performed in autonomous device operation, and/or other functionalities. One or more Extra Infos 527 can be stored in, appended to, or associated with a Collection of Object Representations 525, Instruction Set 526, and/or other element. In some embodiments, the system can obtain Extra Info 527 at a time of creating or generating Collection of Object Representations 525. In other embodiments, the system can obtain Extra Info 527 at a time of acquiring Instruction Set 526. In general, Extra Info 527 can be obtained at any time. Examples of Extra Info 527 include time information, location information, computed information, contextual information, and/or other information. Any information can be utilized that can provide information for enhanced comparisons or decision making performed in autonomous device operation. Which information is utilized and/or stored in Extra Info 527 can be set by a user, by DCADO system administrator, or automatically by the system. Extra Info 527 may include or be referred to as contextual information, and vice versa. Therefore, these terms may be used interchangeably herein depending on context.

In some aspects, time information (i.e. time stamp, etc.) can be utilized and/or stored in Extra Info 527. Time information can be useful in comparisons or decision making performed in autonomous device operation related to a specific time period as Device 98 may be required to perform specific operations at certain parts of day, month, year, and/or other time periods. Time information can be obtained from the system clock, online clock, oscillator, or other time source. In general, Extra Info 527 may include time information related to when Device 98 performed an operation. In other aspects, location information (i.e. coordinates, distance/angle from a known point, address, etc.) can be utilized and/or stored in Extra Info 527. Location information can be useful in comparisons or decision making performed in autonomous device operation related to a specific place as Device 98 may be required to perform specific operations at certain places. Location information can be obtained from a positioning system (i.e. radio signal triangulation, GPS capabilities, etc.), sensors, and/or other location system. In general, Extra Info 527 may include location information related to where Device 98 performed an operation. In further aspects, computed information can be utilized and/or stored in Extra Info 527. Computed information can be useful in comparisons or decision making performed in autonomous device operation where information can be calculated, inferred, or derived from other available information. DCADO Unit 100 and/or other disclosed elements may include computational functionalities to create Extra Info 527 by performing calculations or inferences using other information. In one example, Device's 98 speed can be computed or estimated from Device's 98 location and/or time

information. In another example, Device's 98 bearing (i.e. angle or direction of movement, etc.) can be computed or estimated from Device's 98 location information by utilizing Pythagorean theorem, trigonometry, and/or other theorems, formulas, or disciplines. In a further example, speeds, bearings, distances, and/or other properties of objects around Device 98 can similarly be computed or inferred using known information. In further aspects, observed information can be utilized and/or stored in Extra Info 527. In further aspects, other information can be utilized and/or stored in Extra Info 527. Examples of such other information include user specific information (i.e. skill level, age, gender, etc.), group user information (i.e. access level, etc.), version of Application Program 18, type of Application Program 18, type of Processor 11, type of Logic Circuit 250, type of Device 98, and/or other information.

Referring to Fig. 9, an embodiment where DCADO Unit 100 is part of or operating on Processor 11 is illustrated. In one example, DCADO Unit 100 may be a hardware element or circuit embedded or built into Processor 11. In another example, DCADO Unit 100 may be a program operating on Processor 11.

Referring to Fig. 10, an embodiment where DCADO Unit 100 resides on Server 96 accessible over Network 95 is illustrated. Any number of Devices 98 may connect to such remote DCADO Unit 100 and the remote DCADO Unit 100 may learn their operations in circumstances including objects with various properties. In turn, any number of Devices 98 can utilize the remote DCADO Unit 100 for autonomous operation in circumstances including objects with various properties. A remote DCADO Unit 100 can be offered as a network service (i.e. online application, etc.). In some aspects, a remote DCADO Unit 100 (i.e. global DCADO Unit 100, etc.) may reside on the Internet and be available to all the world's Devices 98 configured to transmit their operations in circumstances including objects with various properties and/or configured to utilize the remote DCADO Unit 100 for autonomous operation in circumstances including objects with various properties. For example, multiple operators (i.e. Users 50, etc.) may operate their Devices 98 where the Devices 98 may be configured to transmit their operations in circumstances including objects with various properties to a remote DCADO Unit 100. Such remote DCADO Unit 100 enables learning of the operators' collective knowledge of operating Device 98 in circumstances including objects with various properties. Server 96 may be or include any type or form of a remote computing device such as an application server, a network service server, a cloud server, a cloud, and/or other remote computing device. Server 96 may include any features, functionalities, and embodiments of the previously described Computing Device 70. It should be understood that Server 96 does not have to be a separate computing device and that Server 96, its elements, or its functionalities can be implemented on Device 98. Network 95 may include various networks, connection types, protocols, interfaces, APIs, and/or other elements or techniques known in art all of which are within the scope of this disclosure. Any of the previously described networks, network or connection types, networking interfaces, and/or other networking elements or techniques can similarly be utilized. Any of the disclosed elements may reside on Server 96 in alternate implementations. In one example, Artificial Intelligence Unit 110 can reside on Server 96 and Acquisition Interface 120 and/or Modification Interface 130 can reside on Device 98. In another example, Knowledgebase 530 can reside on Server 96 and the rest of the elements of DCADO Unit 100 can reside on Device 98. Any other combination of local and remote elements can be implemented.

Referring to Fig. 11, an embodiment of learning and/or using Remote Device's 97 circumstances for autonomous Remote Device 97 operation is illustrated. In such embodiments, in addition to providing input into Object Processing Unit 93 for learning functionalities herein, Sensor 92 (i.e. Camera 92a, Radar 92d, Sonar 92e, etc.) can provide input into Display 21 or other device for User's 50 perception of Remote Device's 97 surrounding.

As User 50 operates Remote Device 97, DCADO Unit 100 may learn Remote Device's 97 operation in circumstances including objects with various properties. Such embodiments can be utilized in any situation where one device controls (i.e. remote controls, etc.) another device, any situation where some or all of the processing is on one device and sensor capabilities are on another device, and/or other situations. In one example, a drone controlling device (i.e. Device 98, etc.) may send control signals to operate a drone (i.e. Remote Device 97, etc.) and receive information on the drone's surrounding from Sensor 92 on the drone. In another example, a robot controlling device (i.e. Device 98, etc.) may send control signals to operate a robot (i.e. Remote Device 97, etc.) and receive information on the robot's surrounding from Sensor 92 on the robot. Any of the disclosed elements in addition to Sensor 92 may reside on Remote Device 97 in alternate implementations.

Referring to Fig. 12, an embodiment of Artificial Intelligence Unit 110 is illustrated. Artificial Intelligence Unit 110 comprises interconnected Knowledge Structuring Unit 520, Knowledgebase 530, Decision-making Unit 540, and Confirmation Unit 550. Other additional elements can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate embodiments.

Artificial Intelligence Unit 110 comprises the functionality for learning Device's 98 operation in circumstances including objects with various properties. Artificial Intelligence Unit 110 comprises the functionality for learning one or more collections of object representations correlated with any instruction sets, data, and/or other information. In some aspects, Artificial Intelligence Unit 110 comprises the functionality for learning one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527. In other aspects, Artificial Intelligence Unit 110 comprises the functionality for learning one or more Collections of Object Representations 525 some of which may not be correlated with any Instruction Sets 526 and/or Extra Info 527. Further, Artificial Intelligence Unit 110 comprises the functionality for anticipating Device's 98 operation in circumstances including objects with various properties. Artificial Intelligence Unit 110 comprises the functionality for anticipating one or more instruction sets, data, and/or other information. Artificial Intelligence Unit 110 comprises the functionality for anticipating one or more Instruction Sets 526 based on one or more incoming Collections of Object Representations 525. Artificial Intelligence Unit 110 comprises the functionality for anticipating one or more Instruction Sets 526 to be used or executed in Device's 98 autonomous operation. Artificial Intelligence Unit 110 also comprises other disclosed functionalities.

Knowledge Structuring Unit 520, Knowledgebase 530, and Decision-making Unit 540 are described later.

Confirmation Unit 550 comprises the functionality for confirming, modifying, evaluating (i.e. rating, etc.), and/or canceling one or more anticipatory Instruction Sets 526, and/or other functionalities. Confirmation Unit 550 is an optional element that can be omitted depending on implementation. In some embodiments, Confirmation Unit 550 can serve as a means of confirming anticipatory Instruction Sets 526. For example, Decision-making Unit 540 may determine one or more anticipatory Instruction Sets 526 and provide them to User 50 for confirmation. User 50 may be provided with an interface (i.e. graphical user interface, selectable list of anticipatory Instruction Sets 526, etc.) to approve or confirm execution of the anticipatory Instruction Sets 526. In some aspects, Confirmation Unit 550 can automate User 50 confirmation. In one example, if one or more incoming Collections of Object Representations 525 from Object Processing Unit 93 and one or more Collections of Object Representations 525 from a Knowledge Cell 800 were found to be a perfect or highly similar match, anticipatory Instruction Sets 526 correlated with the one or more Collections of Object Representations 525 from the Knowledge Cell 800 can be

automatically executed without User's 50 confirmation. Conversely, if one or more incoming Collections of Object Representations 525 from Object Processing Unit 93 and one or more Collections of Object Representations 525 from a Knowledge Cell 800 were found to be less than a highly similar match, anticipatory Instruction Sets 526 correlated with the one or more Collections of Object Representations 525 from the Knowledge Cell 800 can be presented to User 50 for confirmation and/or modifying. Any features, functionalities, and/or embodiments of Similarity Comparison 125 (later described) can be utilized for such similarity determination. In other embodiments, Confirmation Unit 550 can serve as a means of modifying or editing anticipatory Instruction Sets 526. For example, Decision-making Unit 540 may determine one or more anticipatory Instruction Sets 526 and provide them to User 50 for modification. User 50 may be provided with an interface (i.e. graphical user interface, etc.) to modify the anticipatory Instruction Sets 526 before their execution. In further embodiments, Confirmation Unit 550 can serve as a means of evaluating or rating anticipatory Instruction Sets 526 if they matched User's 50 intended operation. For example, Decision-making Unit 540 may determine one or more anticipatory Instruction Sets 526, which the system may automatically execute. User 50 may be provided with an interface (i.e. graphical user interface, etc.) to rate (i.e. on a scale from 0 to 1, etc.) how well Decision-making Unit 540 predicted the executed anticipatory Instruction Sets 526. In some aspects, rating can be automatic and based on a particular function or method that rates how well the anticipatory Instruction Sets 526 matched the desired operation. In one example, a rating function or method can assign a higher rating to anticipatory Instruction Sets 526 that were least modified in the confirmation process. In another example, a rating function or method can assign a higher rating to anticipatory Instruction Sets 526 that were canceled least number of times by User 50. Any other automatic rating function or method can be utilized. In yet other embodiments, Confirmation Unit 550 can serve as a means of canceling anticipatory Instruction Sets 526 if they did not match User's 50 intended operation. For example, Decision-making Unit 540 may determine one or more anticipatory Instruction Sets 526, which the system may automatically execute. The system may save the state of Device 98, Processor 11 (save its register values, etc.), Logic Circuit 250, Application Program 18 (i.e. save its variables, data structures, objects, location of its current instruction, etc.), and/or other processing elements before executing anticipatory Instruction Sets 526. User 50 may be provided with an interface (i.e. graphical user interface, selectable list of prior executed anticipatory Instruction Sets 526, etc.) to cancel one or more of the prior executed anticipatory Instruction Sets 526, and restore Device 98, Processor 11, Logic Circuit 250, Application Program 18, and/or other processing elements to a prior state. In some aspects, Confirmation Unit 550 can optionally be disabled or omitted in order to provide an uninterrupted operation of Device 98, Processor 11, Logic Circuit 250, and/or Application Program 18. For example, a thermostat may be suitable for implementing the user confirmation step, whereas, a vehicle may be less suitable for implementing such interrupting step due to the real time nature of vehicle operation.

Referring to Fig. 13, an embodiment of Knowledge Structuring Unit 520 correlating individual Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527 is illustrated. Knowledge Structuring Unit 520 comprises the functionality for structuring the knowledge of a device's operation in circumstances including objects with various properties, and/or other functionalities. Knowledge Structuring Unit 520 comprises the functionality for correlating one or more Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527. Knowledge Structuring Unit 520 comprises the functionality for creating or generating Knowledge Cell 800 and storing one or more Collections of Object Representations 525 correlated with any

Instruction Sets 526 and/or Extra Info 527 into the Knowledge Cell 800. As such, Knowledge Cell 800 comprises the functionality for storing one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527. Knowledge Cell 800 includes knowledge (i.e. unit of knowledge, etc.) of how Device 98 operated in a circumstance including objects with various properties. Once created or generated, Knowledge Cells 800 can be used in/as neurons, nodes, vertices, or other elements in any of the data structures or arrangements (i.e. neural networks, graphs, sequences, etc.) used for storing the knowledge of Device's 98 operation in circumstances including objects with various properties, thereby facilitating learning functionalities herein. It should be noted that Extra Info 527 may be optionally used in some implementations to enable enhanced comparisons or decision making in autonomous device operation where applicable, and that Extra Info 527 can be omitted in alternate implementations.

In some embodiments, Knowledge Structuring Unit 520 receives one or more Collections of Object Representations 525 from Object Processing Unit 93. Knowledge Structuring Unit 520 may also receive one or more Instruction Sets 526 from Acquisition Interface 120. Knowledge Structuring Unit 520 may further receive any Extra Info 527. Although, Extra Info 527 is not shown in this and/or other figures for clarity of illustration, it should be noted that any Collection of Object Representations 525, Instruction Set 526, and/or other element may include or be associated with Extra Info 527. Knowledge Structuring Unit 520 may correlate one or more Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527. Knowledge Structuring Unit 520 may then create Knowledge Cell 800 and store the one or more Collections of Object Representations 525 correlated with Instruction Sets 526 and/or Extra Info 527 into the Knowledge Cell 800. Knowledge Cell 800 may include any data structure or arrangement that can facilitate such storing. For example, Knowledge Structuring Unit 520 may create Knowledge Cell 800ax and structure within it Collection of Object Representations 525a1 correlated with Instruction Sets 526a1-526a3 and/or any Extra Info 527 (not shown). Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax a Collection of Object Representations 525a2 correlated with Instruction Set 526a4 and/or any Extra Info 527 (not shown). Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax a Collection of Object Representations 525a3 without a correlated Instruction Set 526 and/or Extra Info 527. Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax a Collection of Object Representations 525a4 correlated with Instruction Sets 526a5-526a6 and/or any Extra Info 527 (not shown). Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax a Collection of Object Representations 525a5 without a correlated Instruction Set 526 and/or Extra Info 527. Knowledge Structuring Unit 520 may structure within Knowledge Cell 800ax additional Collections of Object Representations 525 correlated with any number (including zero [i.e. uncorrelated]) of Instruction Sets 526 and/or Extra Info 527 by following similar logic as described above.

In some embodiments, Knowledge Structuring Unit 520 may correlate a Collection of Object Representations 525 with one or more temporally corresponding Instruction Sets 526 and/or Extra Info 527. This way, Knowledge Structuring Unit 520 can structure the knowledge of Device's 98 operation at or around the time of generating Collections of Object Representations 525. Such functionality enables spontaneous or seamless learning of Device's 98 operation in circumstances including objects with various properties as Device 98 is operated in real life situations. In some designs, Knowledge Structuring Unit 520 may receive a stream of Instruction Sets 526 used or executed to effect Device's 98 operations as well as a stream of Collections of Object Representations 525 as the

operations are performed. Knowledge Structuring Unit 520 can then correlate Collections of Object Representations 525 from the stream of Collections of Object Representations 525 with temporally corresponding Instruction Sets 526 from the stream of Instruction Sets 526 and/or any Extra Info 527. Collections of Object Representations 525 without a temporally corresponding Instruction Set 526 may be uncorrelated, for instance. In some aspects, Instruction Sets 526 and/or Extra Info 527 that temporally correspond to a Collection of Object Representations 525 may include Instruction Sets 526 used and/or Extra Info 527 obtained at the time of generating the Collection of Object Representations 525. In other aspects, Instruction Sets 526 and/or Extra Info 527 that temporally correspond to a Collection of Object Representations 525 may include Instruction Sets 526 used and/or Extra Info 527 obtained within a certain time period before and/or after generating the Collection of Object Representations 525. For example, Instruction Sets 526 and/or Extra Info 527 that temporally correspond to a Collection of Object Representations 525 may include Instruction Sets 526 used and/or Extra Info 527 obtained within 50 milliseconds, 1 second, 3 seconds, 20 seconds, 1 minute, 41 minutes, 2 hours, or any other time period before and/or after generating the Collection of Object Representations 525. Such time periods can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. In other aspects, Instruction Sets 526 and/or Extra Info 527 that temporally correspond to a Collection of Object Representations 525 may include Instruction Sets 526 used and/or Extra Info 527 obtained from the time of generating the Collection of Object Representations 525 to the time of generating a next Collection of Object Representations 525. In further aspects, Instruction Sets 526 and/or Extra Info 527 that temporally correspond to a Collection of Object Representations 525 may include Instruction Sets 526 used and/or Extra Info 527 obtained from the time of generating a previous Collection of Object Representations 525 to the time of generating the Collection of Object Representations 525. Any other temporal relationship or correspondence between Collections of Object Representations 525 and correlated Instruction Sets 526 and/or Extra Info 527 can be implemented.

In some embodiments, Knowledge Structuring Unit 520 can structure the knowledge of Device's 98 operation in a circumstance including objects with various properties into any number of Knowledge Cells 800. In some aspects, Knowledge Structuring Unit 520 can structure into a Knowledge Cell 800 a single Collection of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527. In other aspects, Knowledge Structuring Unit 520 can structure into a Knowledge Cell 800 any number (i.e. 2, 4, 7, 17, 29, 87, 1415, 23891, 323674, 8132401, etc.) of Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527. In a special case, Knowledge Structuring Unit 520 can structure all Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 into a single long Knowledge Cell 800. In further aspects, Knowledge Structuring Unit 520 can structure Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 into a plurality of Knowledge Cells 800. In a special case, Knowledge Structuring Unit 520 can store periodic streams of Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 into a plurality of Knowledge Cells 800 such as hourly, daily, weekly, monthly, yearly, or other periodic Knowledge Cells 800.

In some embodiments, Device 98 may include a plurality of Sensors 92 and/or their corresponding Object Processing Units 93. In one example, multiple Sensors 92 may detect objects and/or their properties from different angles or on different sides of Device 98. In another example, one or more Sensors 92 may be placed on different

sub-devices, sub-systems, or elements of Device 98. Using multiple Sensors 92 and/or their corresponding Object Processing Units 93 may provide additional detail in learning and/or using Device's 98 circumstances for autonomous Device 98 operation. In some designs where multiple Sensors 92 and/or their corresponding Object Processing Units 93 are utilized, multiple DCADO Units 100 can also be utilized (i.e. one DCADO Unit 100 for each Sensor 92 and its corresponding Object Processing Unit 93, etc.). In such designs, Collections of Object Representations 525 can be correlated with any Instruction Sets 526 and/or Extra Info 527 as previously described. In other designs where multiple Sensors 92 and/or their corresponding Object Processing Units 93 are utilized, collective Collections of Object Representations 525 from multiple Sensors 92 and their corresponding Object Processing Units 93 can be correlated with any Instruction Sets 526 and/or Extra Info 527.

In some embodiments, Device 98 may include a plurality of Logic Circuits 250, Processors 11, Application Programs 18, and/or other processing elements. For example, each processing element may control a sub-device, sub-system, or an element of Device 98. Using multiple processing elements may provide enhanced control over Device's 98 operation. In some designs where multiple processing elements are utilized, multiple DCADO Units 100 can also be utilized (i.e. one DCADO Unit 100 for each processing element, etc.). In such designs, Collections of Object Representations 525 can be correlated with any Instruction Sets 526 and/or Extra Info 527 as previously described. In other designs where multiple processing elements are utilized, Collections of Object Representations 525 can be correlated with any collective Instruction Sets 526 and/or Extra Info 527 used or executed by a plurality of processing elements.

Any combination of the aforementioned multiple Sensors 92 and/or their corresponding Object Processing Units 93, multiple processing elements, and/or other elements can be implemented in alternate embodiments.

Referring to Fig. 14, another embodiment of Knowledge Structuring Unit 520 correlating individual Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527 is illustrated. In such embodiments, Knowledge Structuring Unit 520 may generate Knowledge Cells 800 each comprising a single Collection of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527.

Referring to Fig. 15, an embodiment of Knowledge Structuring Unit 520 correlating streams of Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527 is illustrated. For example, Knowledge Structuring Unit 520 may create Knowledge Cell 800ax and structure within it a stream of Collections of Object Representations 525a1-525an correlated with Instruction Set 526a1 and/or any Extra Info 527 (not shown). Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax a stream of Collections of Object Representations 525b1-525bn correlated with Instruction Sets 526a2-526a4 and/or and Extra Info 527 (not shown). Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax a stream of Collections of Object Representations 525c1-525cn without correlated Instruction Sets 526 and/or Extra Info 527. Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax a stream of Collections of Object Representations 525d1-525dn correlated with Instruction Sets 526a5-526a6 and/or any Extra Info 527 (not shown). Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax additional streams of Collections of Object Representations 525 correlated with any number (including zero [i.e. uncorrelated]) of Instruction Sets 526 and/or Extra Info 527 by following similar logic as described above. The number of Collections of Object Representations 525 in some or all streams of Collections of Object Representations 525a1-525an, 525b1-525bn, etc. may be equal or different. It should be noted that n or other such letters or indicia may follow the sequence and/or context where

they are indicated. Also, a same letter or indicia such as n may represent a different number in different elements of a drawing.

Referring to Fig. 16, another embodiment of Knowledge Structuring Unit 520 correlating streams of Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527 is illustrated. In such embodiments, Knowledge Structuring Unit 520 may generate Knowledge Cells 800 each comprising a single stream of Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527.

Knowledgebase 530 comprises the functionality for storing the knowledge of a device's operation in circumstances including objects with various properties, and/or other functionalities. Knowledgebase 530 comprises the functionality for storing one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527. Knowledgebase 530 comprises the functionality for storing one or more Knowledge Cells 800 each including one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527. In some aspects, Collections of Object Representations 525 correlated with Instruction Sets 526 and/or Extra Info 527 can be stored directly within Knowledgebase 530 without using Knowledge Cells 800 as the intermediary data structures. In some embodiments, Knowledgebase 530 may be or include Neural Network 530a (later described). In other embodiments, Knowledgebase 530 may be or include Graph 530b (later described). In further embodiments, Knowledgebase 530 may be or include Collection of Sequences 530c (later described). In further embodiments, Knowledgebase 530 may be or include Sequence 533 (later described). In further embodiments, Knowledgebase 530 may be or include Collection of Knowledge Cells 530d (later described). In general, Knowledgebase 530 may be or include any data structure or arrangement capable of storing the knowledge of a device's operation in circumstances including objects with various properties. Knowledgebase 530 may reside locally on Device 98, or remotely (i.e. remote Knowledgebase 530, etc.) on a remote computing device (i.e. server, cloud, etc.) accessible over a network or an interface.

In some embodiments, Knowledgebase 530 from one Device 98 or DCADO Unit 100 can be transferred to one or more other Devices 98 or DCADO Units 100. Therefore, the knowledge of Device's 98 operation in circumstances including objects with various properties learned on one Device 98 or DCADO Unit 100 can be transferred to one or more other Devices 98 or DCADO Units 100. In one example, Knowledgebase 530 can be copied or downloaded to a file or other repository from one Device 98 or DCADO Unit 100 and loaded or inserted into another Device 98 or DCADO Unit 100. In another example, Knowledgebase 530 from one Device 98 or DCADO Unit 100 can be available on a server accessible by other Devices 98 or DCADO Units 100 over a network or an interface. Once loaded into or accessed by a receiving Device 98 or DCADO Unit 100, the receiving Device 98 or DCADO Unit 100 can then implement the knowledge of Device's 98 operation in circumstances including objects with various properties learned on the originating Device 98 or DCADO Unit 100.

In some embodiments, multiple Knowledgebases 530 (i.e. Knowledgebases 530 from different Devices 98 or DCADO Units 100, etc.) can be combined to accumulate collective knowledge of operating Device 98 in circumstances including objects with various properties. In one example, one Knowledgebase 530 can be appended to another Knowledgebase 530 such as appending one Collection of Sequences 530c (later described) to another Collection of Sequences 530c, appending one Sequence 533 (later described) to another Sequence 533, appending one Collection of Knowledge Cells 530d (later described) to another Collection of Knowledge Cells 530d, and/or appending other data structures or elements thereof. In another example, one Knowledgebase 530 can be copied

into another Knowledgebase 530 such as copying one Collection of Sequences 530c into another Collection of Sequences 530c, copying one Collection of Knowledge Cells 530d into another Collection of Knowledge Cells 530d, and/or copying other data structures or elements thereof. In a further example, in the case of Knowledgebase 530 being or including Graph 530b or graph-like data structure (i.e. Neural Network 530a, tree, etc.), a union can be utilized to combine two or more Graphs 530b or graph-like data structures. For instance, a union of two Graphs 530b or graph-like data structures may include a union of their vertex (i.e. node, etc.) sets and their edge (i.e. connection, etc.) sets. Any other operations or combination thereof on graphs or graph-like data structures can be utilized to combine Graphs 530b or graph-like data structures. In a further example, one Knowledgebase 530 can be combined with another Knowledgebase 530 through later described learning processes where Knowledge Cells 800 may be applied one at a time and connected with prior and/or subsequent Knowledge Cells 800 such as in Graph 530b or Neural Network 530a. In such embodiments, instead of Knowledge Cells 800 generated by Knowledge Structuring Unit 520, the learning process may utilize Knowledge Cells 800 from one Knowledgebase 530 to apply them onto another Knowledgebase 530. Any other techniques known in art including custom techniques for combining data structures can be utilized for combining Knowledgebases 530 in alternate implementations. In any of the aforementioned and/or other combining techniques, similarity of elements (i.e. nodes/vertices, edges/connections, etc.) can be utilized in determining whether an element from one Knowledgebase 530 matches an element from another Knowledgebase 530, and substantially or otherwise similar elements may be considered a match for combining purposes in some designs. Any features, functionalities, and embodiments of Similarity Comparison 125 (later described) can be used in such similarity determinations. A combined Knowledgebase 530 can be offered as a network service (i.e. online application, etc.), downloadable file, or other repository to all DCADO Units 100 configured to utilize the combined Knowledgebase 530. For example, a Device 98 including or interfaced with DCADO Unit 100 having access to a combined Knowledgebase 530 can use the collective knowledge learned from multiple Devices 98 for the Device's 98 autonomous operation.

Referring to Fig. 17, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation may include various artificial intelligence models and/or techniques. The disclosed systems, devices, and methods are independent of the artificial intelligence model and/or technique used and any model and/or technique can be utilized to facilitate the functionalities described herein. Examples of these models and/or techniques include deep learning, supervised learning, unsupervised learning, neural networks (i.e. convolutional neural network, recurrent neural network, deep neural network, etc.), search-based, logic and/or fuzzy logic-based, optimization-based, tree/graph/other data structure-based, hierarchical, symbolic and/or sub-symbolic, evolutionary, genetic, multi-agent, deterministic, probabilistic, statistical, and/or other models and/or techniques.

In one example shown in Model A, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation may include a neural network (also referred to as artificial neural network, etc.). As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include a network of Nodes 852 (also referred to as neurons, etc.) and Connections 853 similar to that of a brain. Node 852 can store any data, object, data structure, and/or other item, or reference thereto. Node 852 may also include a function for transforming or manipulating any data, object, data structure, and/or other item. Examples of such transformation

functions include mathematical functions (i.e. addition, subtraction, multiplication, division, sin, cos, log, derivative, integral, etc.), object manipulation functions (i.e. creating an object, modifying an object, deleting an object, appending objects, etc.), data structure manipulation functions (i.e. creating a data structure, modifying a data structure, deleting a data structure, creating a data field, modifying a data field, deleting a data field, etc.), and/or other transformation functions. Connection 853 may include or be associated with a value such as a symbolic label or numeric attribute (i.e. weight, cost, capacity, length, etc.). A computational model can be utilized to compute values from inputs based on a pre-programmed or learned function or method. For example, a neural network may include one or more input neurons that can be activated by inputs. Activations of these neurons can then be passed on, weighted, and transformed by a function to other neurons. Neural networks may range from those with only one layer of single direction logic to multi-layer of multi-directional feedback loops. A neural network can use weights to change the parameters of the network's throughput. A neural network can learn by input from its environment or from self-teaching using written-in rules. A neural network can be utilized as a predictive modeling approach in machine learning. An exemplary embodiment of a neural network (i.e. Neural Network 530a, etc.) is described later.

In another example shown in Model B, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation may include a graph or graph-like data structure. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include Nodes 852 (also referred to as vertices or points, etc.) and Connections 853 (also referred to as edges, arrows, lines, arcs, etc.) organized as a graph. In general, any Node 852 in a graph can be connected to any other Node 852. A Connection 853 may include unordered pair of Nodes 852 in an undirected graph or ordered pair of Nodes 852 in a directed graph. Nodes 852 can be part of the graph structure or external entities represented by indices or references. A graph can be utilized as a predictive modeling approach in machine learning. Nodes 852, Connections 853, and/or other elements or operations of a graph may include any features, functionalities, and embodiments of the aforementioned Nodes 852, Connections 853, and/or other elements or operations of a neural network, and vice versa. An exemplary embodiment of a graph (i.e. Graph 530b, etc.) is described later.

In a further example shown in Model C, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation may include a tree or tree-like data structure. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include Nodes 852 and Connections 853 (also referred to as references, edges, etc.) organized as a tree. In general, a Node 852 in a tree can be connected to any number (i.e. including zero, etc.) of children Nodes 852. A tree can be utilized as a predictive modeling approach in machine learning. Nodes 852, Connections 853, and/or other elements or operations of a tree may include any features, functionalities, and embodiments of the aforementioned Nodes 852, Connections 853, and/or other elements or operations of a neural network and/or graph, and vice versa.

In a further example shown in Model D, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation may include a sequence or sequence-like data structure. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include a structure of Nodes 852 and/or Connections 853 organized as a sequence. In some aspects, Connections 853 may be optionally omitted from a

sequence as the sequential order of Nodes 852 in a sequence may be implied in the structure. A sequence can be utilized as a predictive modeling approach in machine learning. Nodes 852, Connections 853, and/or other elements or operations of a sequence may include any features, functionalities, and embodiments of the aforementioned Nodes 852, Connections 853, and/or other elements or operations of a neural network, graph, and/or tree, and vice versa. An exemplary embodiment of a sequence (i.e. Collection of Sequences 530c, Sequence 533, etc.) is described later.

In yet another example, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation may include a search-based model and/or technique. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include searching through a collection of possible solutions. For example, a search method can search through a neural network, graph, tree, sequence, or other data structure that includes data elements of interest. A search may use heuristics to limit the search for solutions by eliminating choices that are unlikely to lead to the goal. Heuristic techniques may provide a best guess solution. A search can also include optimization. For example, a search may begin with a guess and then refine the guess incrementally until no more refinements can be made. In a further example, the disclosed systems, devices, and methods may include logic-based model and/or technique. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities can use formal or other type of logic. Logic based models may involve making inferences or deriving conclusions from a set of premises. As such, a logic based system can extend existing knowledge or create new knowledge automatically using inferences. Examples of the types of logic that can be utilized include propositional or sentential logic that comprises logic of statements which can be true or false; first-order logic that allows the use of quantifiers and predicates and that can express facts about objects, their properties, and their relations with each other; fuzzy logic that allows degrees of truth to be represented as a value between 0 and 1 rather than simply 0 (false) or 1 (true), which can be used for uncertain reasoning; subjective logic that comprises a type of probabilistic logic that may take uncertainty and belief into account, which can be suitable for modeling and analyzing situations involving uncertainty, incomplete knowledge and different world views; and/or other types of logic. In a further example, the disclosed systems, devices, and methods may include a probabilistic model and/or technique. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities can be implemented to operate with incomplete or uncertain information where probabilities may affect outcomes. Bayesian network, among other models, is an example of a probabilistic tool used for purposes such as reasoning, learning, planning, perception, and/or others. One of ordinary skill in art will understand that the aforementioned artificial intelligence models and/or techniques are described merely as examples of a variety of possible implementations, and that while all possible artificial intelligence models and/or techniques are too voluminous to describe, other artificial intelligence models and/or techniques known in art are within the scope of this disclosure. One of ordinary skill in art will also recognize that an intelligent system may solve a specific problem by using any model and/or technique that works such as, for example, some systems can be symbolic and logical, some can be sub-symbolic neural networks, some can be deterministic or probabilistic, some can be hierarchical, some may include searching techniques, some may include optimization techniques, while

others may use other or a combination of models and/or techniques. In general, any artificial intelligence model and/or technique can be utilized that can facilitate the functionalities described herein.

Referring to Figs. 18A-18C, embodiments of interconnected Knowledge Cells 800 and updating weights of Connections 853 are illustrated. As shown for example in Fig. 18A, Knowledge Cell 800za is connected to Knowledge Cell 800zb and Knowledge Cell 800zc by Connection 853z1 and Connection 853z2, respectively. Each of Connection 853z1 and Connection 853z2 may include or be associated with occurrence count, weight, and/or other parameter or data. The number of occurrences may track or store the number of observations that a Knowledge Cell 800 was followed by another Knowledge Cell 800 indicating a connection or relationship between them. For example, Knowledge Cell 800za was followed by Knowledge Cell 800zb 10 times as indicated by the number of occurrences of Connection 853z1. Also, Knowledge Cell 800za was followed by Knowledge Cell 800zc 15 times as indicated by the number of occurrences of Connection 853z2. The weight of Connection 853z1 can be calculated or determined as the number of occurrences of Connection 853z1 divided by the sum of occurrences of all connections (i.e. Connection 853z1 and Connection 853z2, etc.) originating from Knowledge Cell 800za. Therefore, the weight of Connection 853z1 can be calculated or determined as 10/(10+15)=0.4, for example. Also, the weight of Connection 853z2 can be calculated or determined as 15/(10+15)=0.6, for example. Therefore, the sum of weights of Connection 853z1, Connection 853z2, and/or any other Connections 853 originating from Knowledge Cell 800za may equal to 1 or 100%. As shown for example in Fig. 18B, in the case that Knowledge Cell 800zd is inserted and an observation is made that Knowledge Cell 800zd follows Knowledge Cell 800za, Connection 853z3 can be created between Knowledge Cell 800za and Knowledge Cell 800zd. The occurrence count of Connection 853z3 can be set to 1 and weight determined as 1/(10+15+1)=0.038. The weights of all other connections (i.e. Connection 853z1, Connection 853z2, etc.) originating from Knowledge Cell 800za may be updated to account for the creation of Connection 853z3. Therefore, the weight of Connection 853z1 can be updated as 10/(10+15+1)=0.385. The weight of Connection 853z2 can also be updated as 15/(10+15+1)=0.577. As shown for example in Fig. 18C, in the case that an additional occurrence of Connection 853z1 is observed (i.e. Knowledge Cell 800zb followed Knowledge Cell 800za, etc.), occurrence count of Connection 853z1 and weights of all connections (i.e. Connection 853z1, Connection 853z2, and Connection 853z3, etc.) originating from Knowledge Cell 800za may be updated to account for this observation. The occurrence count of Connection 853z1 can be increased by 1 and its weight updated as 11/(11+15+1)=0.407. The weight of Connection 853z2 can also be updated as 15/(11+15+1)=0.556. The weight of Connection 853z3 can also be updated as 1/(11+15+1)=0.037.

Referring to Fig. 19, an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Collection of Knowledge Cells 530d is illustrated. Collection of Knowledge Cells 530d comprises the functionality for storing any number of Knowledge Cells 800. In some aspects, Knowledge Cells 800 may be stored into or applied onto Collection of Knowledge Cells 530d in a learning or training process. In effect, Collection of Knowledge Cells 530d may store Knowledge Cells 800 that can later be used to enable autonomous Device 98 operation. In some embodiments, Knowledge Structuring Unit 520 structures or generates Knowledge Cells 800 as previously described and the system applies them onto Collection of Knowledge Cells 530d, thereby implementing learning Device's 98 operation in circumstances including objects with various properties. The term apply or applying may refer to storing, copying, inserting, updating, or other similar action, therefore, these terms may be used

interchangeably herein depending on context. The system can perform Similarity Comparisons 125 (later described) of a newly structured Knowledge Cell 800 from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Collection of Knowledge Cells 530d. If a substantially similar Knowledge Cell 800 is not found in Collection of Knowledge Cells 530d, the system may insert (i.e. copy, store, etc.) the Knowledge Cell 800 from Knowledge Structuring Unit 520 into Collection of Knowledge Cells 530d, for example. On the other hand, if a substantially similar Knowledge Cell 800 is found in Collection of Knowledge Cells 530d, the system may optionally omit inserting the Knowledge Cell 800 from Knowledge Structuring Unit 520 as inserting a substantially similar Knowledge Cell 800 may not add much or any additional knowledge to the Collection of Knowledge Cells 530d, for example. Also, inserting a substantially similar Knowledge Cell 800 can optionally be omitted to save storage resources and limit the number of Knowledge Cells 800 that may later need to be processed or compared. Any features, functionalities, and embodiments of Similarity Comparison 125, importance index (later described), similarity index (later described), and/or other disclosed elements can be utilized to facilitate determination of substantial or other similarity and whether to insert a newly structured Knowledge Cell 800 into Collection of Knowledge Cells 530d.

For example, the system can perform Similarity Comparisons 125 (later described) of Knowledge Cell 800ba from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Collection of Knowledge Cells 530d. In the case that a substantially similar match is found between Knowledge Cell 800ba and any of the Knowledge Cells 800 in Collection of Knowledge Cells 530d, the system may perform no action. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bb from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Collection of Knowledge Cells 530d. In the case that a substantially similar match is not found, the system may insert a new Knowledge Cell 800 into Collection of Knowledge Cells 530d and copy Knowledge Cell 800bb into the inserted new Knowledge Cell 800. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bc from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Collection of Knowledge Cells 530d. In the case that a substantially similar match is found between Knowledge Cell 800bc and any of the Knowledge Cells 800 in Collection of Knowledge Cells 530d, the system may perform no action. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bd from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Collection of Knowledge Cells 530d. In the case that a substantially similar match is not found, the system may insert a new Knowledge Cell 800 into Collection of Knowledge Cells 530d and copy Knowledge Cell 800bd into the inserted new Knowledge Cell 800. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800be from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Collection of Knowledge Cells 530d. In the case that a substantially similar match is not found, the system may insert a new Knowledge Cell 800 into Collection of Knowledge Cells 530d and copy Knowledge Cell 800be into the inserted new Knowledge Cell 800. Applying any additional Knowledge Cells 800 from Knowledge Structuring Unit 520 onto Collection of Knowledge Cells 530d follows similar logic or process as the above-described.

Referring to Fig. 20, an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Neural Network 530a is illustrated. Neural Network 530a includes a number of neurons or Nodes 852 interconnected by Connections 853 as previously described. Knowledge Cells 800 are shown instead of Nodes 852 to simplify the illustration as Node 852 includes a Knowledge Cell 800, for example. Therefore, Knowledge Cells 800 and Nodes 852 can be used interchangeably herein depending on context. It should be noted that Node 852 may include other elements

and/or functionalities instead of or in addition to Knowledge Cell 800. In some aspects, Knowledge Cells 800 may be stored into or applied onto Neural Network 530a individually or collectively in a learning or training process. In some designs, Neural Network 530a comprises a number of Layers 854 each of which may include one or more Knowledge Cells 800. Knowledge Cells 800 in successive Layers 854 can be connected by Connections 853. Connection 853 may include or be associated with occurrence count, weight, and/or other parameter or data as previously described. Neural Network 530a may include any number of Layers 854 comprising any number of Knowledge Cells 800. In some aspects, Neural Network 530a may store Knowledge Cells 800 interconnected by Connections 853 where following a path through the Neural Network 530a can later be used to enable autonomous Device 98 operation. It should be understood that, in some embodiments, Knowledge Cells 800 in one Layer 854 of Neural Network 530a need not be connected only with Knowledge Cells 800 in a successive Layer 854, but also in any other Layer 854, thereby creating shortcuts (i.e. shortcut Connections 853, etc.) through Neural Network 530a. A Knowledge Cell 800 can also be connected to itself such as, for example, in recurrent neural networks. In general, any Knowledge Cell 800 can be connected with any other Knowledge Cell 800 anywhere else in Neural Network 530a. In further embodiments, back-propagation of any data or information can be implemented. In one example, back-propagation of similarity (i.e. similarity index, etc.) of compared Knowledge Cells 800 in a path through Neural Network 530a can be implemented. In another example, back-propagation of errors can be implemented. Such back-propagations can then be used to adjust occurrence counts and/or weights of Connections 853 for better future predictions, for example. Any other back-propagation can be implemented for other purposes. Any combination of Nodes 852 (i.e. Nodes 852 comprising Knowledge Cells 800, etc.), Connections 853, Layers 854, and/or other elements or techniques can be implemented in alternate embodiments. Neural Network 530a may include any type or form of a neural network known in art such as a feed-forward neural network, a back-propagating neural network, a recurrent neural network, a convolutional neural network, deep neural network, and/or others including a custom neural network.

In some embodiments, Knowledge Structuring Unit 520 structures or generates Knowledge Cells 800 and the system applies them onto Neural Network 530a, thereby implementing learning Device's 98 operation in circumstances including objects with various properties. The system can perform Similarity Comparisons 125 (later described) of a Knowledge Cell 800 from Knowledge Structuring Unit 520 with Knowledge Cells 800 in a Layer 854 of Neural Network 530a. If a substantially similar Knowledge Cell 800 is not found in the Layer 854 of Neural Network 530a, the system may insert (i.e. copy, store, etc.) the Knowledge Cell 800 from Knowledge Structuring Unit 520 into the Layer 854 of Neural Network 530a, and create a Connection 853 to the inserted Knowledge Cell 800 from a Knowledge Cell 800 in a prior Layer 854 including assigning an occurrence count to the new Connection 853, calculating a weight of the new Connection 853, and updating any other Connections 853 originating from the Knowledge Cell 800 in the prior Layer 854. On the other hand, if a substantially similar Knowledge Cell 800 is found in the Layer 854 of Neural Network 530a, the system may update occurrence count and weight of Connection 853 to that Knowledge Cell 800 from a Knowledge Cell 800 in a prior Layer 854, and update any other Connections 853 originating from the Knowledge Cell 800 in the prior Layer 854.

For example, the system can perform Similarity Comparisons 125 (later described) of Knowledge Cell 800ba from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Layer 854a of Neural Network 530a. In the case that a substantially similar match is found between Knowledge Cell 800ba and Knowledge Cell 800ea, the

system may perform no action since Knowledge Cell 800ea is the initial Knowledge Cell 800. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bb from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Layer 854b of Neural Network 530a. In the case that a substantially similar match is found between Knowledge Cell 800bb and Knowledge Cell 800eb, the system may update occurrence count and weight of Connection 853e1 between Knowledge Cell 800ea and Knowledge Cell 800eb, and update weights of other Connections 853 originating from Knowledge Cell 800ea as previously described. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bc from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Layer 854c of Neural Network 530a. In the case that a substantially similar match is not found, the system may insert Knowledge Cell 800ec into Layer 854c and copy Knowledge Cell 800bc into the inserted Knowledge Cell 800ec. The system may also create Connection 853e2 between Knowledge Cell 800eb and Knowledge Cell 800ec with occurrence count of 1 and weight calculated based on the occurrence count as previously described. The system may also update weights of other Connections 853 (one in this example) originating from Knowledge Cell 800eb as previously described. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bd from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Layer 854d of Neural Network 530a. In the case that a substantially similar match is not found, the system may insert Knowledge Cell 800ed into Layer 854d and copy Knowledge Cell 800bd into the inserted Knowledge Cell 800ed. The system may also create Connection 853e3 between Knowledge Cell 800ec and Knowledge Cell 800ed with occurrence count of 1 and weight of 1. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800be from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Layer 854e of Neural Network 530a. In the case that a substantially similar match is not found, the system may insert Knowledge Cell 800ee into Layer 854e and copy Knowledge Cell 800be into the inserted Knowledge Cell 800ee. The system may also create Connection 853e4 between Knowledge Cell 800ed and Knowledge Cell 800ee with occurrence count of 1 and weight of 1. Applying any additional Knowledge Cells 800 from Knowledge Structuring Unit 520 onto Neural Network 530a follows similar logic or process as the above-described.

Referring now to Similarity Comparison 125, Similarity Comparison 125 comprises the functionality for comparing or matching Knowledge Cells 800 or portions thereof, and/or other functionalities. Similarity Comparison 125 comprises the functionality for comparing or matching Collections of Object Representations 525 or portions thereof. Similarity Comparison 125 comprises the functionality for comparing or matching streams of Collections of Object Representations 525 or portions thereof. Similarity Comparison 125 comprises the functionality for comparing or matching Object Representations 625 or portions thereof. Similarity Comparison 125 comprises the functionality for comparing or matching Object Properties 630 or portions thereof. Similarity Comparison 125 comprises the functionality for comparing or matching Instruction Sets 526, Extra Info 527, text (i.e. characters, words, phrases, etc.), numbers, and/or other elements or portions thereof. Similarity Comparison 125 may include functions, rules, and/or logic for performing matching or comparisons and for determining that while a perfect match is not found, a partial or similar match has been found. In some aspects, a partial match may include a substantially or otherwise similar match, and vice versa. Therefore, these terms may be used interchangeably herein depending on context. As such, Similarity Comparison 125 may include determining substantial similarity or substantial match of compared elements. Although, substantial similarity or substantial match is frequently used herein, it should be understood that any level of similarity, however high or low, may be utilized as defined by the rules (i.e. thresholds, etc.) for similarity.

The rules for similarity or similar match can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. In some designs, Similarity Comparison 125 comprises the functionality to automatically define appropriately strict rules for determining similarity of the compared elements. Similarity Comparison 125 can therefore set, reset, and/or adjust the strictness of the rules for finding or determining similarity of the compared elements, thereby fine tuning Similarity Comparison 125 so that the rules for determining similarity are appropriately strict. In some aspects, the rules for determining similarity may include a similarity threshold. As such, Similarity Comparison 125 can determine similarity of compared elements if their similarity exceeds a similarity threshold. In other aspects, the rules for determining similarity may include a difference threshold. As such, Similarity Comparison 125 can determine similarity of compared elements if their difference is lower than a difference threshold. In further aspects, the rules for determining similarity may include other thresholds. Similarity Comparison 125 enables comparing circumstances including objects with various properties and determining their similarity or match. In one example, a circumstance including an object detected at a distance of 8m and an angle/bearing of 64° relative to Device 98 may be found similar or matching by Similarity Comparison 125 to a circumstance including the same or similar object detected at a distance of 8.6m and an angle/bearing of 59° relative to Device 98. In another example, a circumstance including an object detected as a passenger vehicle may be found similar or matching by Similarity Comparison 125 to a circumstance including an object detected as a sport utility vehicle. In general, any one or more properties (i.e. existence, type, identity, distance, bearing/angle, location, shape/size, activity, etc.) of one or more objects can be utilized for determining similarity or match of circumstances including objects with various properties. Therefore, Similarity Comparison 125 provides flexibility in comparing and determining similarity of a variety of possible circumstances of Device 98.

In some embodiments where compared Knowledge Cells 800 include a single Collection of Object Representations 525, in determining similarity of Knowledge Cells 800, Similarity Comparison 125 can perform comparison of individual Collections of Object Representations 525 or portions (i.e. Object Representations 625, Object Properties 630, etc.) thereof such as comparison of Collection of Object Representations 525 or portions thereof from one Knowledge Cell 800 with Collection of Object Representations 525 or portions thereof from another Knowledge Cell 800. In some aspects, total equivalence is achieved when Collection of Object Representations 525 or portions thereof from one Knowledge Cell 800 matches Collection of Object Representations 525 or portions thereof from another Knowledge Cell 800. If total equivalence is not found, Similarity Comparison 125 may attempt to determine substantial or other similarity of compared Knowledge Cells 800.

In some embodiments, in determining substantial similarity of individually compared Collections of Object Representations 525 (i.e. Collections of Object Representations 525 from the compared Knowledge Cells 800, etc.), Similarity Comparison 125 can compare one or more Object Representations 625 or portions (i.e. Object Properties 630, etc.) thereof from one Collection of Object Representations 525 with one or more Object Representations 625 or portions thereof from another Collection of Object Representations 525. In some aspects, total equivalence is found when all Object Representations 625 or portions thereof from one Collection of Object Representations 525 match all Object Representations 625 or portions thereof from another Collection of Object Representations 525. In other aspects, if total equivalence is not found, Similarity Comparison 125 may attempt to determine substantial similarity of compared Collections of Object Representations 525. In one example, substantial similarity can be

achieved when most of the Object Representations 625 or portions thereof from the compared Collections of Object Representations 525 match or substantially match. In another example, substantial similarity can be achieved when at least a threshold number (i.e. 1, 2, 4, 7, 18, etc.) or percentage (i.e. 41%, 62%, 79%, 85%, 93%, etc.) of Object Representations 625 or portions thereof from the compared Collections of Object Representations 525 match or substantially match. Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching Object Representations 625 or portions thereof from the compared Collections of Object Representations 525 exceeds a threshold number (i.e. 1, 2, 4, 7, 18, etc.) or a threshold percentage (i.e. 41%, 62%, 79%, 85%, 93%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or percentage of Object Representations 625 or portions thereof from the compared Collections of Object Representations 525 match or substantially match. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In some aspects, Similarity Comparison 125 can utilize the importance (i.e. as indicated by importance index [later described], etc.) of Object Representations 625 or portions thereof for determining substantial similarity of Collections of Object Representations 525. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to more important Object Representations 625 or portions thereof such as Object Representations 625 representing near Objects 615, Object Representations 625 representing large Objects 615, etc., thereby tolerating mismatches in less important Object Representations 625 or portions thereof such as Object Representations 625 representing distant Objects 615, Object Representations 625 representing small Objects 615, etc. In general, any Object Representation 625 or portion thereof can be assigned higher or lower importance. In further aspects, Similarity Comparison 125 can omit some of the Object Representations 625 or portions thereof from the comparison in determining substantial similarity of Collections of Object Representations 525. In one example, Object Representations 625 representing distant Objects 615 can be omitted from comparison. In another example, Object Representations 625 representing small Objects 615 can be omitted from comparison. In general, any Object Representation 625 or portion thereof can be omitted from comparison depending on implementation.

Similarity Comparison 125 can automatically adjust (i.e. increase or decrease) the strictness of the rules for determining substantial similarity of Collections of Object Representations 525. In some aspects, such adjustment in strictness can be done by Similarity Comparison 125 in response to determining that total equivalence of compared Collections of Object Representations 525 had not been found. Similarity Comparison 125 can keep adjusting the strictness rules until a substantial similarity is found. All the rules or settings of substantial similarity can be set, reset, or adjusted by Similarity Comparison 125 in response to another strictness level determination. For example, Similarity Comparison 125 may attempt to find a match or substantial match in a certain percentage (i.e. 81%, etc.) of Object Representations 625 or portions thereof from the compared Collections of Object Representations 525. If the comparison does not determine substantial similarity of compared Collections of Object Representations 525, Similarity Comparison 125 may decide to decrease the strictness of the rules. In response, Similarity Comparison 125 may attempt to find fewer matching or substantially matching Object Representations 625 or portions thereof than in the previous attempt using stricter rules. If the comparison still does not determine substantial similarity of compared Collections of Object Representations 525, Similarity Comparison 125 may further decrease the strictness (i.e. down to a certain minimum strictness or threshold, etc.) by requiring fewer Object Representations 625 or

portions thereof to match or substantially match, thereby further increasing a chance of finding substantial similarity in compared Collections of Object Representations 525.

Where a reference to Object Representation 625 is used herein it should be understood that a portion of Object Representation 625 (i.e. Object Property 630, etc.) or a plurality of Object Representations 625 can be used instead of or in addition to the Object Representation 625. In one example, instead of or in addition to Object Representation 625, Object Properties 630 and/or other portions that constitute an Object Representation 625 can be compared. In another example, instead of or in addition to Object Representation 625, plurality of Object Representations 625 can be compared. As such, any operations, rules, logic, and/or functions operating on Object Representation 625 may similarly apply to any portion of Object Representation 625 and/or a plurality of Object Representations 625 as applicable. In general, whole Object Representations 625, portions of Object Representations 625, and/or pluralities of Object Representations 625, including any operations thereon, can be combined to arrive at desired results. Some or all of the above-described rules, logic, and/or techniques can be utilized alone or in combination with each other or with other rules, logic, and/or techniques. One of ordinary skill in art will recognize that other techniques known in art for determining similarity of Object Representations 625 and/or other data that would be too voluminous to describe are within the scope of this disclosure.

In some embodiments, in determining substantial similarity of Object Representations 625 (i.e. Object Representations 625 from the compared Collections of Object Representations 525, etc.), Similarity Comparison 125 can compare Object Properties 630 or portions (i.e. characters, words, numbers, etc.) thereof from one Object Representation 625 with Object Properties 630 or portions thereof from another Object Representation 625. In some aspects, total equivalence is found when all Object Properties 630 or portions thereof of one Object Representation 625 match all Object Properties 630 or portions thereof of another Object Representation 625. In other aspects, if total equivalence is not found, Similarity Comparison 125 may attempt to determine substantial similarity of compared Object Representations 625. In one example, substantial similarity can be achieved when most of the Object Properties 630 or portions thereof from the compared Object Representations 625 match or substantially match. In another example, substantial similarity can be achieved when at least a threshold number (i.e. 1, 2, 3, 6, 11, etc.) or percentage (i.e. 55%, 61%, 78%, 82%, 99%, etc.) of Object Properties 630 or portions thereof from the compared Object Representations 625 match or substantially match. Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching Object Properties 630 or portions thereof from the compared Object Representations 625 exceeds a threshold number (i.e. 1, 2, 3, 6, 11, etc.) or a threshold percentage (i.e. 55%, 61%, 78%, 82%, 99%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or percentage of Object Properties 630 or portions thereof from the compared Object Representations 625 match or substantially match. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In further aspects, Similarity Comparison 125 can utilize Categories 635 associated with Object Properties 630 for determining substantial similarity of Object Representations 625. In one example, Object Properties 630 or portions thereof from the compared Object Representations 625 in a same Category 635 may be compared. This way, Object Properties 630 or portions thereof can be compared with their own peers. In one instance, Object Properties 630 or portions thereof from the compared Object Representations 625 in Category 635 "Type" may be compared. Any text comparison technique can be utilized in such comparing. In another

instance, Object Properties 630 or portions thereof from the compared Object Representations 625 in Category 635 “Distance” or “Bearing” may be compared. Any number comparison technique can be utilized in such comparing. In a further instance, Object Properties 630 or portions thereof from the compared Object Representations 625 in Category 635 “Shape” may be compared. Any model, point cloud, or other computer construct comparison technique can be utilized in such comparing. In further aspects, Similarity Comparison 125 can utilize the importance (i.e. as indicated by importance index [later described], etc.) of Object Properties 630 or portions thereof for determining substantial similarity of Object Representations 625. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to more important Object Properties 630 or portions thereof such as Object Properties 630 or portions thereof in Categories 635 “Type”, “Distance”, “Bearing”, etc., thereby tolerating mismatches in less important Object Properties 630 or portions thereof such as Object Properties 630 or portions thereof in Categories 635 “Identity”, “Shape”, etc. In general, any Object Property 630 or portion thereof can be assigned higher or lower importance. In further aspects, Similarity Comparison 125 can omit some of the Object Properties 630 or portions thereof from the comparison in determining substantial similarity of Object Representations 625. In one example, Object Properties 630 or portions thereof in Category 635 “Identity” can be omitted from comparison. In another example, Object Properties 630 or portions thereof in Category 635 “Shape” can be omitted from comparison. In general, any Object Property 630 or portion thereof can be omitted from comparison depending on implementation.

Similarity Comparison 125 can automatically adjust (i.e. increase or decrease) the strictness of the rules for determining substantial similarity of Object Representations 625. In some aspects, such adjustment in strictness can be done by Similarity Comparison 125 in response to determining that total equivalence of compared Object Representations 625 had not been found. Similarity Comparison 125 can keep adjusting the strictness rules until a substantial similarity is found. All the rules or settings of substantial similarity can be set, reset, or adjusted by Similarity Comparison 125 in response to another strictness level determination. For example, Similarity Comparison 125 may attempt to find a match or substantial match in a certain percentage (i.e. 87%, etc.) of Object Properties 630 or portions thereof from the compared Object Representations 625. If the comparison does not determine substantial similarity of compared Object Representations 625, Similarity Comparison 125 may decide to decrease the strictness of the rules. In response, Similarity Comparison 125 may attempt to find fewer matching or substantially matching Object Properties 630 or portions thereof than in the previous attempt using stricter rules. If the comparison still does not determine substantial similarity of compared Object Representations 625, Similarity Comparison 125 may further decrease the strictness (i.e. down to a certain minimum strictness or threshold, etc.) by requiring fewer Object Properties 630 or portions thereof to match or substantially match, thereby further increasing a chance of finding substantial similarity in compared Object Representations 625. In further aspects, an adjustment in strictness can be done by Similarity Comparison 125 in response to determining that multiple substantially similar Object Representations 625 had been found. Similarity Comparison 125 can keep adjusting the strictness of the rules until a best of the substantially similar Object Representations 625 is found. For example, Similarity Comparison 125 may attempt to find a match or substantial match in a certain percentage (i.e. 65%, etc.) of Object Properties 630 or portions thereof from the compared Object Representations 625. If the comparison determines a number of substantially similar Object Representations 625, Similarity Comparison 125 may decide to increase the strictness of the rules to decrease the number of substantially similar Object Representations 625. In response,

Similarity Comparison 125 may attempt to find more matching or substantially matching Object Properties 630 or portions thereof in addition to the earlier found Object Properties 630 or portions thereof to limit the number of substantially similar Object Representations 625. If the comparison still provides more than one substantially similar Object Representation 625, Similarity Comparison 125 may further increase the strictness by requiring additional Object Properties 630 or portions thereof to match or substantially match, thereby further narrowing the number of substantially similar Object Representations 625 until a best substantially similar Object Representation 625 is found.

Where a reference to Object Property 630 is used herein it should be understood that a portion of Object Property 630 or a plurality of Object Properties 630 can be used instead of or in addition to the Object Property 630. In one example, instead of or in addition to Object Property 630, characters, words, numbers, and/or other portions that constitute an Object Property 630 can be compared. In another example, instead of or in addition to Object Property 630, a plurality of Object Properties 630 can be compared. As such, any operations, rules, logic, and/or functions operating on Object Property 630 may similarly apply to any portion of Object Property 630 and/or a plurality of Object Properties 630 as applicable. In general, whole Object Properties 630, portions of Object Properties 630, and/or pluralities of Object Properties 630, including any operations thereon, can be combined to arrive at desired results. Some or all of the above-described rules, logic, and/or techniques can be utilized alone or in combination with each other or with other rules, logic, and/or techniques. One of ordinary skill in art will recognize that other techniques known in art for determining similarity of Object Properties 630 and/or other data that would be too voluminous to describe are within the scope of this disclosure.

In some embodiments where compared Knowledge Cells 800 include a stream of Collections of Object Representations 525, in determining similarity of Knowledge Cells 800, Similarity Comparison 125 can perform collective comparison of Collections of Object Representations 525 or portions (i.e. Object Representations 625, Object Properties 630, etc.) thereof such as comparison of a stream of Collections of Object Representations 525 or portions thereof from one Knowledge Cell 800 with a stream of Collections of Object Representations 525 or portions thereof from another Knowledge Cell 800. Similarity Comparison 125 of collectively compared Collections of Object Representations 525 or portions thereof may include any features, functionalities, and embodiments of the previously described Similarity Comparison 125 of individually compared Collections of Object Representations 525 or portions thereof. In some aspects, total equivalence is found when all Collections of Object Representations 525 or portions thereof from one Knowledge Cell 800 match all Collections of Object Representations 525 or portions thereof from another Knowledge Cell 800. If total equivalence is not found, Similarity Comparison 125 may attempt to determine substantial or other similarity of compared Knowledge Cells 800. In one example, substantial similarity can be achieved when most of the Collections of Object Representations 525 or portions thereof from the compared Knowledge Cells 800 match or substantially match. In another example, substantial similarity can be achieved when at least a threshold number (i.e. 1, 2, 4, 9, 33, 138, etc.) or percentage (i.e. 39%, 58%, 77%, 88%, 94%, etc.) of Collections of Object Representations 525 or portions thereof from the compared Knowledge Cells 800 match or substantially match. Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching Collections of Object Representations 525 or portions thereof from the compared Knowledge Cells 800 exceeds a threshold number (i.e. 1, 2, 4, 9, 33, 138, etc.) or a threshold percentage (i.e. 39%, 58%, 77%, 88%, 94%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or

percentage of Collections of Object Representations 525 or portions thereof from the compared Knowledge Cells 800 match or substantially match. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In some aspects, Similarity Comparison 125 can utilize the importance (i.e. as indicated by importance index [later described], etc.) of Collections of Object Representations 525 or portions thereof for determining substantial similarity of Knowledge Cells 800. In one example, substantial similarity can be achieved when matches or substantial matches are found with respect to more important Collections of Object Representations 525 or portions thereof such as more substantive or larger Collections of Object Representations 525 (i.e. Collections of Object Representations 525 comprising a higher number of Object Representations 625, etc.) or portions thereof, etc., thereby tolerating mismatches in less important Collections of Object Representations 525 or portions thereof such as less substantive or smaller Collections of Object Representations 525 (i.e. Collections of Object Representations 525 comprising a lower number of Object Representations 625, etc.) or portions thereof, etc. In general, any Collection of Object Representations 525 or portion thereof can be assigned higher or lower importance. In other aspects, Similarity Comparison 125 can utilize the order of Collections of Object Representations 525 or portions thereof for determining substantial similarity of Knowledge Cells 800. In one example, substantial similarity can be achieved when matches or substantial matches are found in earlier Collections of Object Representations 525 or portions thereof from the compared Knowledge Cells 800, thereby tolerating mismatches in later Collections of Object Representations 525 or portions thereof. In another example, substantial similarity can be achieved when matches or substantial matches are found in corresponding (i.e. similarly ordered, temporally related, etc.) Collections of Object Representations 525 or portions thereof from the compared Knowledge Cells 800. In one instance, a 94th Collection of Object Representations 525 or portions thereof from one Knowledge Cell 800 can be compared with a 94th Collection of Object Representations 525 or portions thereof from another Knowledge Cell 800. In another instance, a 94th Collection of Object Representations 525 or portions thereof from one Knowledge Cell 800 can be compared with a number of Collections of Object Representations 525 or portions thereof around (i.e. preceding and/or following) a 94th Collection of Object Representations 525 from another Knowledge Cell 800. This way, flexibility can be implemented in finding a substantially similar Collection of Object Representations 525 or portions thereof if the Collections of Object Representations 525 or portions thereof in the compared Knowledge Cells 800 are not perfectly aligned. In a further instance, Similarity Comparison 125 can utilize Dynamic Time Warping (DTW) and/or other techniques known in art for comparing and/or aligning temporal sequences (i.e. streams of Collections of Object Representations 525 or portions thereof, etc.) that may vary in time or speed. In further aspects, Similarity Comparison 125 can omit some of the Collections of Object Representations 525 or portions thereof from the comparison in determining substantial similarity of Knowledge Cells 800. In one example, less substantive or smaller Collections of Object Representations 525 or portions thereof can be omitted from comparison. In another example, some or all Collections of Object Representations 525 or portions thereof related to a specific time period can be omitted from comparison. In general, any Collection of Object Representations 525 or portion thereof can be omitted from comparison depending on implementation.

Similarity Comparison 125 can automatically adjust (i.e. increase or decrease) the strictness of the rules for determining substantial similarity of Knowledge Cells 800. In some aspects, such adjustment in strictness can be done by Similarity Comparison 125 in response to determining that total equivalence of compared Knowledge Cells

800 had not been found. Similarity Comparison 125 can keep adjusting the strictness of the rules until substantial similarity is found. All the rules or settings of substantial similarity can be set, reset, or adjusted by Similarity Comparison 125 in response to another strictness level determination. For example, Similarity Comparison 125 may attempt to find a match or substantial match in a certain percentage (i.e. 92%, etc.) of Collections of Object Representations 525 or portions thereof from the compared Knowledge Cells 800. If the comparison does not determine substantial similarity of compared Knowledge Cells 800, Similarity Comparison 125 may decide to decrease the strictness of the rules. In response, Similarity Comparison 125 may attempt to find fewer matching or substantially matching Collections of Object Representations 525 or portions thereof than in the previous attempt using stricter rules. If the comparison still does not determine substantial similarity of compared Knowledge Cells 800, Similarity Comparison 125 may further decrease (i.e. down to a certain minimum strictness or threshold, etc.) the strictness by requiring fewer Collections of Object Representations 525 or portions thereof to match or substantially match, thereby further increasing a chance of finding substantial similarity in compared Knowledge Cells 800. In further aspects, an adjustment in strictness can be done by Similarity Comparison 125 in response to determining that multiple substantially similar Knowledge Cells 800 had been found. Similarity Comparison 125 can keep adjusting the strictness of the rules until a best of the substantially similar Knowledge Cells 800 is found. For example, Similarity Comparison 125 may attempt to find a match or substantial match in a certain percentage (i.e. 71%, etc.) of Collections of Object Representations 525 or portions thereof from the compared Knowledge Cells 800. If the comparison determines a number of substantially similar Knowledge Cells 800, Similarity Comparison 125 may decide to increase the strictness of the rules to decrease the number of substantially similar Knowledge Cells 800. In response, Similarity Comparison 125 may attempt to find more matching or substantially matching Collections of Object Representations 525 or portions thereof in addition to the earlier found Collections of Object Representations 525 or portions thereof to limit the number of substantially similar Knowledge Cells 800. If the comparison still provides more than one substantially similar Knowledge Cell 800, Similarity Comparison 125 may further increase the strictness by requiring additional Collections of Object Representations 525 or portions thereof to match or substantially match, thereby further narrowing the number of substantially similar Knowledge Cells 800 until a best substantially similar Knowledge Cell 800 is found.

Some or all of the aforementioned rules, logic, and/or techniques for determining substantial similarity of Knowledge Cells 800 can be utilized alone or in combination with each other or with other rules, logic, and/or techniques. One of ordinary skill in art will recognize that other techniques known in art for determining similarity of Knowledge Cells 800 and/or other data that would be too voluminous to describe are within the scope of this disclosure.

In any of the comparisons involving numbers such as, for example, Object Properties 630 including numbers (i.e. distances, bearings/angles, etc.), Similarity Comparison 125 can compare a number from one Object Property 630 with a number from another Object Property 630. In some aspects, total equivalence is found when the number from one Object Property 630 equals the number from another Object Property 630. In other aspects, if total equality is not found, Similarity Comparison 125 may attempt to determine substantial similarity of the compared numbers using a tolerance or threshold for determining a match. In some aspects, Similarity Comparison 125 can utilize a threshold for acceptable number difference in determining a match of compared numbers. For example, a threshold for acceptable number difference (i.e. absolute difference, etc.) can be set at 10. Therefore, 130 matches

or is sufficiently similar to 135 because the number difference (i.e. 5 in this example) is lower than the threshold for acceptable number difference (i.e. 10 in this example, etc.). Furthermore, 130 does not match or is not sufficiently similar to 143 because the number difference (i.e. 13 in this example) is greater than the threshold for acceptable number difference. Any other threshold for acceptable number difference can be used such as 0.024, 1, 8, 15, 77, 197, 2438, 728322, and/or others. In other aspects, Similarity Comparison 125 can utilize a threshold for acceptable percentage difference in determining a match of compared numbers. For example, a threshold for acceptable percentage difference can be set at 10%. Therefore, 100 matches or is sufficiently similar to 106 because the percentage difference (i.e. 6% in this example) is lower than the threshold for acceptable percentage difference (i.e. 10% in this example). Furthermore, 100 does not match or is not sufficiently similar to 84 because the percentage difference (i.e. 16% in this example) is higher than the threshold for acceptable percentage difference. Any other threshold for acceptable percentage difference can be used such as 0.68%, 1%, 3%, 11%, 33%, 69%, 87%, and/or others. The aforementioned thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. Other techniques known in art for comparing numbers can be utilized herein.

In any of the comparisons involving text such as, for example, Object Properties 630 including text (i.e. types, identities, etc.), Similarity Comparison 125 can compare words, characters, and/or other text from one Object Property 630 with words, characters, and/or other text from another Object Property 630. In some aspects, total equivalence is found when all words, characters, and/or other text from one Object Property 630 match all words, characters, and/or other text from another Object Property 630. In other aspects, if total equivalence is not found, Similarity Comparison 125 may attempt to determine substantial similarity of compared Object Properties 630. In one example, substantial similarity can be achieved when most of the words, characters, and/or other text from the compared Object Properties 630 match or substantially match. In another example, substantial similarity can be achieved when at least a threshold number (i.e. 1, 2, 3, 4, 7, 11, etc.) or percentage (i.e. 38%, 63%, 77%, 84%, 98%, etc.) of words, characters, and/or other text from the compared Object Properties 630 match or substantially match. Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching words, characters, and/or other text from the compared Object Properties 630 exceeds a threshold number (i.e. 1, 2, 3, 4, 7, 11, etc.) or a threshold percentage (i.e. 48%, 63%, 77%, 84%, 98%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or percentage of words, characters, and/or other text from the compared Object Properties 630 match or substantially match. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In further aspects, Similarity Comparison 125 can utilize the importance (i.e. as indicated by importance index [later described], etc.) of words, characters, and/or other text for determining substantial similarity of Object Properties 630. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to more important words, characters, and/or other text such as longer words and/or other text, thereby tolerating mismatches in less important words, characters, and/or other text such as shorter words and/or other text. In general, any word, character, and/or other text can be assigned higher or lower importance. In further aspects, Similarity Comparison 125 can utilize the order of words, characters, and/or other text for determining substantial similarity of Object Properties 630. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to front-most

words, characters, and/or other text, thereby tolerating mismatches in later words, characters, and/or other text. In further aspects, Similarity Comparison 125 can utilize semantic conversion to account for variations of words and/or other text. In one example, Object Property 630 may include a word “house”. In addition to searching for the exact word in a compared Object Property 630, Similarity Comparison 125 can employ semantic conversion and attempt to match “home”, “residence”, “dwelling”, “place”, or other semantically similar variations of the word with a meaning “house”. In another example, Object Property 630 may include a word “buy”. In addition to searching for the exact word in a compared Object Property 630, Similarity Comparison 125 can employ semantic conversion and attempt to match “buying”, “bought”, or other semantically similar variations of the word with a meaning “buy” in different tenses. Any other grammatical analysis or transformation can be utilized to cover the full scope of word and/or other text variations. In some designs, semantic conversion can be implemented using a thesaurus or dictionary. In another example, semantic conversion can be implemented using a table where each row comprises semantically similar variations of a word and/or other text. In further aspects, Similarity Comparison 125 can utilize a language model for understanding or interpreting the concepts contained in the words and/or other text and compare the concepts instead of or in addition to the words and/or other text. Examples of language models include unigram model, n-gram model, neural network language model, bag of words model, and/or others. Any of the techniques for matching of words can similarly be used for matching of concepts. In further aspects, Similarity Comparison 125 can omit some of the words, characters, and/or other text from the comparison in determining substantial similarity of Object Properties 630. In one example, rear-most words, characters, and/or other text can be omitted from comparison. In another example, shorter words and/or other text can be omitted from comparison. In general, any word, character, and/or other text can be omitted from comparison depending on implementation. Other techniques known in art for comparing words, characters, and/or other text can be utilized herein.

In some embodiments, Similarity Comparison 125 can compare one or more Extra Info 527 (i.e. time information, location information, computed information, contextual information, and/or other information, etc.) in addition to or instead of comparing Collections of Object Representations 525 or portions thereof in determining substantial similarity of Knowledge Cells 800. Extra Info 527 can be set to be less, equally, or more important (i.e. as indicated by importance index [later described], etc.) than Collections of Object Representations 525, Object Representations 625, Object Properties 630, and/or other elements in the comparison. Since Extra Info 527 may include any contextual or other information that can be useful in determining similarity of any compared elements, Extra Info 527 can be used to enhance any of the aforementioned similarity determinations as applicable.

In some embodiments, Similarity Comparison 125 can also compare one or more Instruction Sets 526 in addition to or instead of comparing Collections of Object Representations 525 or portions thereof in determining substantial similarity of Knowledge Cells 800. In some aspects, Similarity Comparison 125 can compare portions of Instruction Sets 526 to determine substantial or other similarity of Instruction Sets 526. Similar to the above-described thresholds, thresholds for the number or percentage of matching portions of the compared Instruction Sets 526 can be utilized in determining substantial or other similarity of the compared Instruction Sets 526. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In other aspects, Similarity Comparison 125 can compare text (i.e. characters, words, phrases, etc.), numbers, or other data (i.e. bits, etc.) to determine substantial or other similarity of Instruction Sets 526. Any other comparison technique can be

utilized in comparing Instruction Sets 526 in alternate implementations. Instruction Sets 526 can be set to be less, equally, or more important (i.e. as indicated by importance index [later described], etc.) than Collections of Object Representations 525, Object Representations 625, Object Properties 630, Extra Info 527, and/or other elements in the comparison.

In some embodiments, an importance index (not shown) or other importance ranking technique can be used in any of the previously described comparisons or other processing involving elements of different importance. Importance index indicates importance of the element to or with which the index is assigned or associated. For example, importance index may indicate importance of a Knowledge Cell 800, Collection of Object Representations 525, Object Representation 625, Object Property 630, Instruction Set 526, Extra Info 527, and/or other element to or with which the index is assigned or associated. In some aspects, importance index on a scale from 0 to 1 can be utilized, although, any other range can also be utilized. Importance index can be stored in or associated with the element to which the index pertains. Importance indexes of various elements can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. In one example, a higher Importance index can be assigned to more substantive or larger Collections of Object Representations 525 (i.e. Collections of Object Representations 525 comprising a higher number of Object Representations 625, etc.). In another example, a higher importance index can be assigned to Object Representations 625 representing closer, larger, and/or other Objects 615. Any importance index can be assigned to or associated with any element described herein depending on implementation. Any importance ranking technique can be utilized as or instead of importance index in alternate embodiments.

In some embodiments, Similarity Comparison 125 may generate a similarity index (not shown) for any of the compared elements. Similarity index indicates how well an element is matched with another element. For example, similarity index indicates how well a Knowledge Cell 800, Collection of Object Representations 525, Object Representation 625, Object Property 630, Instruction Set 526, Extra Info 527, and/or other element is matched with a compared element. In some aspects, similarity index on a scale from 0 to 1 can be utilized, although, any other range can also be utilized. Similarity index can be generated by Similarity Comparison 125 whether substantial or other similarity between the compared elements is achieved or not. In one example, similarity index can be determined for a Knowledge Cell 800 based on a ratio/percentage of matched or substantially matched Collections of Object Representations 525 relative to the number of Collections of Object Representations 525 in the compared Knowledge Cell 800. Specifically, similarity index of 0.91 is determined if 91% of Collections of Object Representations 525 of one Knowledge Cell 800 match or substantially match Collections of Object Representations 525 of another Knowledge Cell 800. In some designs, importance (i.e. as indicated by importance index, etc.) of one or more Collections of Object Representations 525 can be included in the calculation of a weighted similarity index. Similar determination of similarity index can be implemented with Collections of Object Representations 525, Object Representations 625, Object Properties 630, Instruction Sets 526, Extra Info 527, and/or other elements or portions thereof. Any combination of the aforementioned similarity index determinations or calculations can be utilized in alternate embodiments. Any similarity ranking technique can be utilized to determine or calculate similarity index in alternate embodiments.

Referring to Fig. 21, an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Neural Network 530a comprising shortcut Connections 853 is illustrated. In some designs, Knowledge Cells 800 in one Layer 854 of Neural Network 530a can be connected with Knowledge Cells 800 in any Layer 854, not only in a successive Layer 854, thereby creating shortcuts (i.e. shortcut Connections 853, etc.) through Neural Network 530a. In some aspects, creating a shortcut Connection 853 can be implemented by performing Similarity Comparisons 125 of a Knowledge Cell 800 from Knowledge Structuring Unit 520 with Knowledge Cells 800 in any Layer 854 when applying (i.e. storing, copying, etc.) the Knowledge Cell 800 from Knowledge Structuring Unit 520 onto Neural Network 530a. Once created, shortcut Connections 853 enable a wider variety of Knowledge Cells 800 to be considered when selecting a path through Neural Network 530a. In some embodiments, Knowledge Structuring Unit 520 structures or generates Knowledge Cells 800 and the system applies them onto Neural Network 530a, thereby implementing learning Device's 98 operation in circumstances including objects with various properties. The system can perform Similarity Comparisons 125 of a Knowledge Cell 800 from Knowledge Structuring Unit 520 with Knowledge Cells 800 in one or more Layers 854 of Neural Network 530a. If a substantially similar Knowledge Cell 800 is not found in the one or more Layers 854 of Neural Network 530a, the system may insert (i.e. copy, store, etc.) the Knowledge Cell 800 from Knowledge Structuring Unit 520 into a Layer 854 of Neural Network 530a, and create a Connection 853 to the inserted Knowledge Cell 800 from a prior Knowledge Cell 800 including assigning an occurrence count to the new Connection 853, calculating a weight of the new Connection 853, and updating any other Connections 853 originating from the prior Knowledge Cell 800. On the other hand, if a substantially similar Knowledge Cell 800 is found in the one or more Layers 854 of Neural Network 530a, the system may update occurrence count and weight of Connection 853 to that Knowledge Cell 800 from a prior Knowledge Cell 800, and update any other Connections 853 originating from the prior Knowledge Cell 800. Any of the previously described and/or other techniques for comparing, inserting, updating, and/or other operations on Knowledge Cells 800, Connections 853, Layers 854, and/or other elements can similarly be utilized in Neural Network 530a that comprises shortcut Connections 853.

Referring to Fig. 22, an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Graph 530b is illustrated. In some aspects, any Knowledge Cell 800 can be connected with any other Knowledge Cell 800 in Graph 530b. In other aspects, any Knowledge Cell 800 can be connected with itself and/or any other Knowledge Cell 800 in Graph 530b. In some embodiments, Knowledge Structuring Unit 520 structures or generates Knowledge Cells 800 and the system applies (i.e. store, copy, etc.) them onto Graph 530b, thereby implementing learning Device's 98 operation in circumstances including objects with various properties. The system can perform Similarity Comparisons 125 of a Knowledge Cell 800 from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Graph 530b. If a substantially similar Knowledge Cell 800 is not found in Graph 530b, the system may insert (i.e. copy, store, etc.) the Knowledge Cell 800 from Knowledge Structuring Unit 520 into Graph 530b, and create a Connection 853 to the inserted Knowledge Cell 800 from a prior Knowledge Cell 800 including assigning an occurrence count to the new Connection 853, calculating a weight of the new Connection 853, and updating any other Connections 853 originating from the prior Knowledge Cell 800. On the other hand, if a substantially similar Knowledge Cell 800 is found in Graph 530b, the system may update occurrence count and weight of Connection 853 to that Knowledge Cell 800 from a prior Knowledge Cell 800, and update any other Connections 853 originating from the prior

Knowledge Cell 800. Any of the previously described and/or other techniques for comparing, inserting, updating, and/or other operations on Knowledge Cells 800, Connections 853, and/or other elements can similarly be utilized in Graph 530b.

For example, the system can perform Similarity Comparisons 125 of Knowledge Cell 800ba from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Graph 530b. In the case that a substantially similar match is not found, the system may insert Knowledge Cell 800ha into Graph 530b and copy Knowledge Cell 800ba into the inserted Knowledge Cell 800ha. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bb from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Graph 530b. In the case that a substantially similar match is found between Knowledge Cell 800bb and Knowledge Cell 800hb, the system may create Connection 853h1 between Knowledge Cell 800ha and Knowledge Cell 800hb with occurrence count of 1 and weight of 1. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bc from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Graph 530b. In the case that a substantially similar match is found between Knowledge Cell 800bc and Knowledge Cell 800hc, the system may update occurrence count and weight of Connection 853h2 between Knowledge Cell 800hb and Knowledge Cell 800hc, and update weights of other outgoing Connections 853 (one in this example) originating from Knowledge Cell 800hb as previously described. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bd from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Graph 530b. In the case that a substantially similar match is not found, the system may insert Knowledge Cell 800hd into Graph 530b and copy Knowledge Cell 800bd into the inserted Knowledge Cell 800hd. The system may also create Connection 853h3 between Knowledge Cell 800hc and Knowledge Cell 800hd with occurrence count of 1 and weight calculated based on the occurrence count as previously described. The system may also update weights of other outgoing Connections 853 (one in this example) originating from Knowledge Cell 800hc as previously described. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800be from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Graph 530b. In the case that a substantially similar match is not found, the system may insert Knowledge Cell 800he into Graph 530b and copy Knowledge Cell 800be into the inserted Knowledge Cell 800he. The system may also create Connection 853h4 between Knowledge Cell 800hd and Knowledge Cell 800he with occurrence count of 1 and weight of 1. Applying any additional Knowledge Cells 800 from Knowledge Structuring Unit 520 onto Graph 530b follows similar logic or process as the above-described.

Referring to Fig. 23, an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Collection of Sequences 530c is illustrated. Collection of Sequences 530c comprises the functionality for storing one or more Sequences 533. Sequence 533 comprises the functionality for storing any number of Knowledge Cells 800. For example, Knowledge Structuring Unit 520 structures or generates Knowledge Cells 800 and the system applies them onto Collection of Sequences 530c, thereby implementing learning Device's 98 operation in circumstances including objects with various properties. The system can perform collective Similarity Comparisons 125 of Knowledge Cells 800 from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Sequences 533 of Collection of Sequences 530c to find a Sequence 533 comprising Knowledge Cells 800 that are collectively substantially similar to the Knowledge Cells 800 from Knowledge Structuring Unit 520. If Sequence 533 comprising such collectively substantially similar Knowledge Cells 800 is not found in Collection of Sequences 530c, the system

may create a new Sequence 533 comprising the Knowledge Cells 800 from Knowledge Structuring Unit 520 and insert (i.e. copy, store, etc.) the new Sequence 533 into Collection of Sequences 530c. On the other hand, if Sequence 533 comprising collectively substantially similar Knowledge Cells 800 is found in Collection of Sequences 530c, the system may optionally omit inserting the Knowledge Cells 800 from Knowledge Structuring Unit 520 into Collection of Sequences 530c as inserting a similar Sequence 533 may not add much or any additional knowledge. This approach can save storage resources and limit the number of Knowledge Cells 800 that may later need to be processed or compared. In some aspects, a Sequence 533 may include Knowledge Cells 800 relating to a single operation of Device 98. In other aspects, a Sequence 533 may include Knowledge Cells 800 relating to a part of an operation of Device 98. In further aspects, one or more long Sequences 533 each including Knowledge Cells 800 of multiple operations of Device 98 can be utilized. In one example, Knowledge Cells 800 of all operations can be stored in a single long Sequence 533 in which case Collection of Sequences 530c as a separate element can be omitted. In another example, Knowledge Cells 800 of multiple operations can be included in a plurality of long Sequences 533 such as hourly, daily, weekly, monthly, yearly, or other periodic or other Sequences 533. Similarity Comparisons 125 can be performed by traversing the one or more long Sequences 533 to find a match or substantially similar match. For instance, the system can perform collective Similarity Comparisons 125 of Knowledge Cells 800 from Knowledge Structuring Unit 520 with Knowledge Cells 800 in subsequences of a long Sequence 533 in incremental or other traversing pattern to find a subsequence comprising Knowledge Cells 800 that are collectively substantially similar to the Knowledge Cells 800 from Knowledge Structuring Unit 520. The incremental traversing pattern may start from one end of a long Sequence 533 and move the comparison subsequence up or down one or any number of incremental Knowledge Cells 800 at a time. Other traversing patterns or methods can be employed such as starting from the middle of the Sequence 533 and subdividing the resulting sub-sequences in a recursive pattern, or any other traversing pattern or method. If a subsequence comprising collectively substantially similar Knowledge Cells 800 is not found in the long Sequence 533, the system may concatenate or append the Knowledge Cells 800 from Knowledge Structuring Unit 520 to the long Sequence 533. In further aspects, Connections 853 can optionally be used in Sequence 533 to connect Knowledge Cells 800. For example, a Knowledge Cell 800 can be connected not only with a next Knowledge Cell 800 in the Sequence 533, but also with any other Knowledge Cell 800 in the Sequence 533, thereby creating alternate routes or shortcuts through the Sequence 533. Any number of Connections 853 connecting any Knowledge Cells 800 can be utilized. Any of the previously described and/or other techniques for comparing, inserting, updating, and/or other operations on Knowledge Cells 800, Connections 853, and/or other elements can similarly be utilized in Sequences 533 and/or Collection of Sequences 530c.

In some embodiments, various elements and/or techniques can be utilized in the aforementioned substantial similarity determinations with respect to collectively compared Knowledge Cells 800 and/or other elements. In some aspects, substantial similarity of collectively compared Knowledge Cells 800 can be determined based on similarities or similarity indexes of the individually compared Knowledge Cells 800. In one example, an average of similarities or similarity indexes of individually compared Knowledge Cells 800 can be used to determine similarity of collectively compared Knowledge Cells 800. In another example, a weighted average of similarities or similarity indexes of individually compared Knowledge Cells 800 can be used to determine similarity of collectively compared Knowledge Cells 800. For instance, to affect the weighting of collective similarity, a higher weight or

importance (i.e. importance index, etc.) can be assigned to the similarities or similarity indexes of some Knowledge Cells 800 and lower for other Knowledge Cells 800. Any higher or lower weight or importance assignment can be implemented. In other aspects, any of the previously described or other thresholds for substantial similarity of individually compared elements can similarly be utilized for collectively compared elements. In one example, substantial similarity of collectively compared Knowledge Cells 800 can be achieved when their collective similarity or similarity index exceeds a similarity threshold. In another example, substantial similarity of collectively compared Knowledge Cells 800 can be achieved when at least a threshold number or percentage of Knowledge Cells 800 from the collectively compared Knowledge Cells 800 match or substantially match. Similarly, substantial similarity of collectively compared Knowledge Cells 800 can be achieved when a number or percentage of matching or substantially matching Knowledge Cells 800 from the collectively compared Knowledge Cells 800 exceeds a threshold. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. Similar elements and/or techniques as the aforementioned can be used for similarity determinations of other collectively compared elements such as Collections of Object Representations 525, Object Representations 625, Object Properties 630, Instruction Sets 526, Extra Info 527, and/or others. Similarity determinations of collectively compared elements may include any features, functionalities, and embodiments of Similarity Comparison 125, and vice versa.

Any of the previously described data structures or arrangements of Knowledge Cells 800 such as Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, and/or others can be used alone, or in combination with each other or with other elements, in alternate embodiments. In one example, a path in Neural Network 530a or Graph 530b may include its own separate sequence of Knowledge Cells 800 that are not interconnected with Knowledge Cells 800 in other paths. In another example, a part of a path in Neural Network 530a or Graph 530b may include a sequence of Knowledge Cells 800 interconnected with Knowledge Cells 800 in other paths, whereas, another part of the path may include its own separate sequence of Knowledge Cells 800 that are not interconnected with Knowledge Cells 800 in other paths. Any other combinations or arrangements of Knowledge Cells 800 can be implemented.

Referring to Fig. 24, an embodiment of determining anticipatory Instruction Sets 526 from a single Knowledge Cell 800 is illustrated. Knowledge Cell 800 may be part of a Knowledgebase 530 (i.e. Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.) such as Collection of Knowledge Cells 530d. Decision-making Unit 540 comprises the functionality for anticipating or determining a device's operation in circumstances including objects with various properties. Decision-making Unit 540 comprises the functionality for anticipating or determining Instruction Sets 526 to be used or executed in Device's 98 autonomous operation. In some aspects, Instruction Sets 526 anticipated or determined to be used or executed in Device's 98 autonomous operation may be referred to as anticipatory Instruction Sets 526, alternate Instruction Sets 526, and/or other suitable name or reference. Therefore, these terms can be used interchangeably herein depending on context. Decision-making Unit 540 also comprises other disclosed functionalities.

In some aspects, Decision-making Unit 540 may anticipate or determine Instruction Sets 526 (i.e. anticipatory Instruction Sets 526, etc.) for autonomous Device 98 operation by performing Similarity Comparisons 125 of incoming Collections of Object Representations 525 or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Knowledgebase 530 (i.e.

Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.). A Knowledge Cell 800 includes knowledge (i.e. one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527, etc.) of how Device 98 operated in a circumstance including objects with various properties as previously described. When one or more Collections of Object Representations 525 representing objects with similar properties are received in the future, Decision-making Unit 540 can anticipate the Instruction Sets 526 (i.e. anticipatory Instruction Sets 526, etc.) previously learned in a similar circumstance, thereby enabling autonomous Device 98 operation. In some aspects, Decision-making Unit 540 can perform Similarity Comparisons 125 of incoming Collections of Object Representations 525 from Object Processing Unit 93 with Collections of Object Representations 525 from Knowledge Cells 800 in Knowledgebase 530 (i.e. Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.). If one or more substantially similar Collections of Object Representations 525 or portions thereof are found in a Knowledge Cell 800 from Knowledgebase 530, Instruction Sets 526 (i.e. anticipatory Instruction Sets 526, etc.) for autonomous Device 98 operation can be anticipated in Instruction Sets 526 correlated with the one or more Collections of Object Representations 525 from the Knowledge Cell 800. In some designs, subsequent one or more Instruction Sets 526 for autonomous Device 98 operation can be anticipated in Instruction Sets 526 correlated with subsequent Collections of Object Representations 525 from the Knowledge Cell 800 or other Knowledge Cells 800, thereby anticipating not only current, but also additional future Instruction Sets 526. Although, Extra Info 527 is not shown in this and/or other figures for clarity of illustration, it should be noted that any Collection of Object Representations 525, Instruction Set 526, and/or other element may include or be associated with Extra Info 527 and that Decision-making Unit 540 can utilize Extra Info 527 for enhanced decision making.

For example, Decision-making Unit 540 can perform Similarity Comparison 125 of Collection of Object Representations 525l1 or portions thereof from Object Processing Unit 93 with Collection of Object Representations 525a1 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a1 or portions thereof from Knowledge Cell 800oa may be found substantially similar. Decision-making Unit 540 can anticipate Instruction Sets 526a1-526a3 correlated with Collection of Object Representations 525a1, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparison 125 of Collection of Object Representations 525l2 or portions thereof from Object Processing Unit 93 with Collection of Object Representations 525a2 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a2 or portions thereof from Knowledge Cell 800oa may be found substantially similar. Decision-making Unit 540 can anticipate Instruction Set 526a4 correlated with Collection of Object Representations 525a2, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparison 125 of Collection of Object Representations 525l3 or portions thereof from Object Processing Unit 93 with Collection of Object Representations 525a3 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a3 or portions thereof from Knowledge Cell 800oa may be found substantially similar. Decision-making Unit 540 may not anticipate any Instruction Sets 526 since none are correlated with Collection of Object Representations 525a3. Decision-making Unit 540 can then perform Similarity Comparison 125 of Collection of Object Representations 525l4 or portions thereof from Object Processing Unit 93 with Collection of Object Representations 525a4 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a4 or portions thereof from Knowledge Cell 800oa may not be found substantially similar. Decision-making Unit 540 can then

perform Similarity Comparison 125 of Collection of Object Representations 525l5 or portions thereof from Object Processing Unit 93 with Collection of Object Representations 525a5 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a5 or portions thereof from Knowledge Cell 800oa may not be found substantially similar. Decision-making Unit 540 can implement similar logic or process for any additional Collections of Object Representations 525 from Object Processing Unit 93, and so on.

It should be understood that any of the described elements and/or techniques in the foregoing example can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells 800 or elements (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof would be affected accordingly. In one example, Extra Info 527 can be included in the Similarity Comparisons 125 as previously described. In another example, as history of incoming Collections of Object Representations 525 becomes available, Decision-making Unit 540 can perform collective Similarity Comparisons 125 of the history of Collections of Object Representations 525 or portions thereof from Object Processing Unit 93 with subsequences of Collections of Object Representations 525 or portions thereof from Knowledge Cell 800. In a further example, the described comparisons in a single Knowledge Cell 800 may be performed on any number of Knowledge Cells 800 sequentially or in parallel. Parallel processors such as a plurality of Processors 11 or cores thereof can be utilized for such parallel processing. In a further example, various arrangements of Collections of Object Representations 525 and/or other elements in a Knowledge Cell 800 can be utilized as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to Fig. 25, an embodiment of determining anticipatory Instruction Sets 526 by traversing a single Knowledge Cell 800 is illustrated. Knowledge Cell 800 may be part of a Knowledgebase 530 (i.e. Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.) such as Collection of Knowledge Cells 530d. For example, Decision-making Unit 540 can perform Similarity Comparison 125 of Collection of Object Representations 525l1 or portions thereof from Object Processing Unit 93 with Collection of Object Representations 525a1 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a1 or portions thereof from Knowledge Cell 800oa may not be found substantially similar. Decision-making Unit 540 can then perform Similarity Comparison 125 of Collection of Object Representations 525l1 or portions thereof from Object Processing Unit 93 with Collection of Object Representations 525a2 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a2 or portions thereof from Knowledge Cell 800oa may not be found substantially similar. Decision-making Unit 540 can then perform Similarity Comparison 125 of Collection of Object Representations 525l1 or portions thereof from Object Processing Unit 93 with Collection of Object Representations 525a3 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a3 or portions thereof from Knowledge Cell 800oa may be found substantially similar. Decision-making Unit 540 may not anticipate any Instruction Sets 526 since none are correlated with Collection of Object Representations 525a3. Decision-making Unit 540 can then perform Similarity Comparison 125 of Collection of Object Representations 525l2 or portions thereof from Object Processing Unit 93 with Collection of Object Representations 525a4 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a4 or portions thereof from Knowledge Cell 800oa may be found substantially similar. Decision-making Unit 540 can anticipate Instruction Sets 526a5-

526a6 correlated with Collection of Object Representations 525a4, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparison 125 of Collection of Object Representations 525l3 or portions thereof from Object Processing Unit 93 with Collection of Object Representations 525a5 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a5 or portions thereof from Knowledge Cell 800oa may be found substantially similar. Decision-making Unit 540 may not anticipate any Instruction Sets 526 since none are correlated with Collection of Object Representations 525a5. Decision-making Unit 540 can implement similar logic or process for any additional Collections of Object Representations 525 from Object Processing Unit 93, and so on.

It should be understood that any of the described elements and/or techniques in the foregoing example can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells 800 or elements (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof would be affected accordingly. In one example, Extra Info 527 can be included in the Similarity Comparisons 125 as previously described. In another example, as history of incoming Collections of Object Representations 525 becomes available, Decision-making Unit 540 can perform collective Similarity Comparisons 125 of the history of Collections of Object Representations 525 or portions thereof from Object Processing Unit 93 with subsequences of Collections of Object Representations 525 or portions thereof from Knowledge Cell 800. In a further example, traversing may be performed in incremental traversing pattern such as starting from one end of Knowledge Cell 800 and moving the comparison subsequence up or down the list one or any number of incremental Collections of Object Representations 525 at a time. Other traversing patterns or methods can be employed such as starting from the middle of the Knowledge Cell 800 and subdividing the resulting subsequence in a recursive pattern, or any other traversing pattern or method. In a further example, the described traversing of a single Knowledge Cell 800 may be performed on any number of Knowledge Cells 800 sequentially or in parallel. Parallel processors such as a plurality of Processors 11 or cores thereof can be utilized for such parallel processing. In a further example, various arrangements of Collections of Object Representations 525 and/or other elements in a Knowledge Cell 800 can be utilized as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to Fig. 26, an embodiment of determining anticipatory Instruction Sets 526 using collective similarity comparisons is illustrated. For example, Decision-making Unit 540 can perform Similarity Comparisons 125 of Collection of Object Representations 525l1 or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Collection of Knowledge Cells 530d. Collection of Object Representations 525c1 or portions thereof from Knowledge Cell 800rc may be found substantially similar with highest similarity. Decision-making Unit 540 can anticipate any Instruction Sets 526 (not shown) correlated with Collection of Object Representations 525c1, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform collective Similarity Comparisons 125 of Collections of Object Representations 525l1-525l2 or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Collection of Knowledge Cells 530d. Collections of Object Representations 525c1-525c2 or portions thereof from Knowledge Cell 800rc may be found

substantially similar with highest similarity. Decision-making Unit 540 can anticipate any Instruction Sets 526 (not shown) correlated with Collection of Object Representations 525c2, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform collective Similarity Comparisons 125 of Collections of Object Representations 525l1-525l3 or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Collection of Knowledge Cells 530d. Collections of Object Representations 525d1-525d3 or portions thereof from Knowledge Cell 800rd may be found substantially similar with highest similarity. Decision-making Unit 540 can anticipate any Instruction Sets 526 (not shown) correlated with Collection of Object Representations 525d3, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform collective Similarity Comparisons 125 of Collections of Object Representations 525l1-525l4 or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Collection of Knowledge Cells 530d. Collections of Object Representations 525d1-525d4 or portions thereof from Knowledge Cell 800rd may be found substantially similar with highest similarity. Decision-making Unit 540 can anticipate any Instruction Sets 526 (not shown) correlated with Collection of Object Representations 525d4, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform collective Similarity Comparisons 125 of Collections of Object Representations 525l1-525l5 or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Collection of Knowledge Cells 530d. Collections of Object Representations 525d1-525d5 or portions thereof from Knowledge Cell 800rd may be found substantially similar with highest similarity. Decision-making Unit 540 can anticipate any Instruction Sets 526 (not shown) correlated with Collection of Object Representations 525d5, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can implement similar logic or process for any additional Collections of Object Representations 525 from Object Processing Unit 93, and so on.

In some embodiments, various elements and/or techniques can be utilized in the aforementioned similarity determinations with respect to collectively compared Collections of Object Representations 525 and/or other elements. In some aspects, similarity of collectively compared Collections of Object Representations 525 can be determined based on similarities or similarity indexes of the individually compared Collections of Object Representations 525. In one example, an average of similarities or similarity indexes of individually compared Collections of Object Representations 525 can be used to determine similarity of collectively compared Collections of Object Representations 525. In another example, a weighted average of similarities or similarity indexes of individually compared Collections of Object Representations 525 can be used to determine similarity of collectively compared Collections of Object Representations 525. For instance, to affect the weighting of collective similarity, a higher weight or importance (i.e. importance index, etc.) can be assigned to the similarities or similarity indexes of some (i.e. more substantive or larger, etc.) Collections of Object Representations 525 and lower for other (i.e. less substantive or smaller, etc.) Collections of Object Representations 525. Any other higher or lower weight or importance assignment can be implemented. In other aspects, any of the previously described or other thresholds for substantial similarity of individually compared elements can be similarly utilized for collectively compared elements. In one example, substantial similarity of collectively compared Collections of Object Representations 525 can be achieved when their collective similarity or similarity index exceeds a similarity threshold. In another example, substantial similarity of collectively compared Collections of Object Representations 525 can be achieved when at

least a threshold number or percentage of Collections of Object Representations 525 or portions thereof from the collectively compared Collections of Object Representations 525 match or substantially match. Similarly, substantial similarity of collectively compared Collections of Object Representations 525 can be achieved when a number or percentage of matching or substantially matching Collections of Object Representations 525 or portions thereof from the collectively compared Collections of Object Representations 525 exceeds a threshold. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. Similar elements and/or techniques as the aforementioned can be used for similarity determinations of other collectively compared elements such as Object Representations 625, Object Properties 630, Instruction Sets 526, Extra Info 527, Knowledge Cells 800, and/or others. Similarity determinations of collectively compared elements may include any features, functionalities, and embodiments of Similarity Comparison 125, and vice versa.

It should be understood that any of the described elements and/or techniques in the foregoing example can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells 800 or elements (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info 527 in Similarity Comparisons 125, traversing of Knowledge Cells 800 or other elements, using history of Collections of Object Representations 525 or Knowledge Cells 800 for collective Similarity Comparisons 125, using various arrangements of Collections of Object Representations 525 and/or other elements in a Knowledge Cell 800, and/or others can similarly be utilized in this example. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to Fig. 27, an embodiment of determining anticipatory Instruction Sets 526 using Neural Network 530a is illustrated. In some aspects, determining anticipatory Instruction Sets 526 using Neural Network 530a may include selecting a path of Knowledge Cells 800 or elements (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof through Neural Network 530a. Decision-making Unit 540 can utilize various elements and/or techniques for selecting a path through Neural Network 530a. Although, these elements and/or techniques are described with respect to Neural Network 530a below, they can similarly be used in any Knowledgebase 530 (i.e. Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.) as applicable.

In some embodiments, Decision-making Unit 540 can utilize similarity index in selecting Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof in a path through Neural Network 530a. For instance, similarity index may indicate how well one Knowledge Cell 800 or portions thereof are matched with another Knowledge Cell 800 or portions thereof as previously described. In one example, Decision-making Unit 540 may select a Knowledge Cell 800 comprising one or more Collections of Object Representations 525 with highest similarity index even if Connection 853 pointing to that Knowledge Cell 800 has less than the highest weight. Therefore, similarity index or other such element or parameter can override or disregard the weight of a Connection 853 or other element. In another example, Decision-making Unit 540 may select a Knowledge Cell 800 comprising one or more Collections of Object Representations 525 whose similarity

index is higher than or equal to a weight of Connection 853 pointing to that Knowledge Cell 800. In a further example, Decision-making Unit 540 may select a Knowledge Cell 800 comprising one or more Collections of Object Representations 525 whose similarity index is lower than or equal to a weight of Connection 853 pointing to that Knowledge Cell 800. Similarity index can be set to be more, less, or equally important than a weight of a Connection 853.

In some embodiments, Decision-making Unit 540 can utilize Connections 853 in selecting Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof in a path through Neural Network 530a. In some aspects, Decision-making Unit 540 can take into account weights of Connections 853 among the interconnected Knowledge Cells 800 in choosing from which Knowledge Cell 800 to compare one or more Collections of Object Representations 525 first, second, third, and so on. Specifically, for instance, Decision-making Unit 540 can perform Similarity Comparisons 125 with one or more Collections of Object Representations 525 from Knowledge Cell 800 pointed to by the highest weight Connection 853 first, Collections of Object Representations 525 from Knowledge Cell 800 pointed to by the second highest weight Connection 853 second, and so on. In other aspects, Decision-making Unit 540 can stop performing Similarity Comparisons 125 as soon as it finds one or more substantially similar Collections of Object Representations 525 in an interconnected Knowledge Cell 800. In further aspects, Decision-making Unit 540 may only follow the highest weight Connection 853 to arrive at a Knowledge Cell 800 comprising one or more Collections of Object Representations 525 to be compared, thereby disregarding Connections 853 with less than the highest weight. In further aspects, Decision-making Unit 540 may ignore weights and/or other parameters of Connections 853. In further aspects, Decision-making Unit 540 may ignore Connections 853.

In some embodiments, Decision-making Unit 540 can utilize a bias to adjust similarity index, weight of a Connection 853, and/or other element or parameter used in selecting Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof in a path through Neural Network 530a. In one example, Decision-making Unit 540 may select a Knowledge Cell 800 comprising one or more Collections of Object Representations 525 whose similarity index multiplied by or adjusted for a bias is higher than or equal to a weight of Connection 853 pointing to that Knowledge Cell 800. In another example, Decision-making Unit 540 may select a Knowledge Cell 800 comprising one or more Collections of Object Representations 525 whose similarity index multiplied by or adjusted for a bias is lower than or equal to a weight of Connection 853 pointing to that Knowledge Cell 800. In a further example, bias can be used to resolve deadlock situations where similarity index is equal to a weight of a Connection 853. In some aspects, bias can be expressed in percentages such as 0.3 percent, 1.2 percent, 25.7 percent, 79.8 percent, 99.9 percent, 100.1 percent, 155.4 percent, 298.6 percent, 1105.5 percent, and so on. For example, a bias below 100 percent decreases an element or parameter to which it is applied, a bias equal to 100 percent does not change the element or parameter to which it is applied, and a bias higher than 100 percent increases the element or parameter to which it is applied. In general, any amount of bias can be utilized depending on implementation. Bias can be applied to one or more of a weight of a Connection 853, similarity index, any other element or parameter, and/or all or any combination of them. Also, different biases can be applied to each of a weight of a Connection 853, similarity index, or any other element or parameter. For example, 30 percent bias can be applied to similarity index and 15 percent bias can be applied to a weight of a Connection 853. Also, different biases can be applied to various Layers 854 of Neural Network 530a, and/or other

disclosed elements. Bias can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input.

Any other element and/or technique can be utilized in selecting Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof in a path through Neural Network 530a.

In some embodiments, Neural Network 530a may include knowledge (i.e. interconnected Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527, etc.) of how Device 98 operated in circumstances including objects with various properties. In some aspects, determining anticipatory Instruction Sets 526 using Neural Network 530a may include selecting a path of Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof through Neural Network 530a. Individual and/or collective Similarity Comparisons 125 can be used to determine substantial similarity of the individually and/or collectively compared Collections of Object Representations 525 or portions thereof. Substantial similarity may be used primarily for selecting a path through Neural Network 530a, whereas, weight of any Connection 853 may be used secondarily or not at all.

For example, Decision-making Unit 540 can perform Similarity Comparisons 125 of Collections of Object Representations 525a1-525an or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from one or more Knowledge Cells 800 in Layer 854a (or any other one or more Layers 854, etc.). Collections of Object Representations 525 or portions thereof from Knowledge Cell 800ta may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object Representations 525b1-525bn or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from one or more Knowledge Cells 800 in Layer 854b interconnected with Knowledge Cell 800ta. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800tb may be found collectively substantially similar with highest similarity, thus, Decision-making Unit 540 may follow Connection 853t1 disregarding its less than highest weight. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Since Connection 853t2 is the only connection from Knowledge Cell 800tb, Decision-making Unit 540 may follow Connection 853t2 and perform Similarity Comparisons 125 of Collections of Object Representations 525c1-525cn or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cell 800tc in Layer 854c. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800tc may be found collectively substantially similar. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object

Representations 525d1-525dn or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from one or more Knowledge Cells 800 in Layer 854d interconnected with Knowledge Cell 800tc. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800td may be found collectively substantially similar with highest similarity, thus, Decision-making Unit 540 may follow Connection 853t3. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object Representations 525e1-525en or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from one or more Knowledge Cells 800 in Layer 854e interconnected with Knowledge Cell 800td. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800te may be found collectively substantially similar with highest similarity, thus, Decision-making Unit 540 may follow Connection 853t4. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can implement similar logic or process for any additional Collections of Object Representations 525 from Object Processing Unit 93, and so on.

The foregoing exemplary embodiment provides an example of utilizing a combination of collective Similarity Comparisons 125, individual Similarity Comparisons 125, Connections 853, and/or other elements or techniques. It should be understood that any of these elements and/or techniques can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof in a path through Neural Network 530a would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info 527 in Similarity Comparisons 125, traversing of Knowledge Cells 800 or other elements, using history of Collections of Object Representations 525 or Knowledge Cells 800 for collective Similarity Comparisons 125, using various arrangements of Collections of Object Representations 525 and/or other elements in a Knowledge Cell 800, and/or others can similarly be utilized in this example. These elements and/or techniques can similarly be utilized in Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, and/or other data structures or arrangements. In some aspects, instead of anticipating Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525, Decision-making Unit 540 can anticipate instruction Sets 526 correlated with substantially similar streams of Collections of Object Representations 525. In other aspects, any time that substantial similarity or other similarity threshold is not achieved in compared Collections of Object Representations 525 or portions thereof from any of the Knowledge Cells 800, Decision-making Unit 540 can decide to look for a substantially or otherwise similar Collections of Object Representations 525 or portions thereof in Knowledge Cells 800 elsewhere in Neural Network 530a such as in any Layer 854 subsequent to a current Layer 854, in the first Layer 854, in the entire Neural Network 530a, and/or others, even if such Knowledge Cell 800 may be unconnected with a prior Knowledge Cell 800. It should be noted that any of Collections of Object Representations 525a1-525an, Collections of Object Representations 525b1-525bn, Collections of Object Representations 525c1-525cn, Collections of Object

Representations 525d1-525dn, Collections of Object Representations 525e1-525en, etc. may include one Collection of Object Representations 525 or a stream of Collections of Object Representations 525. It should also be noted that any Knowledge Cell 800 may include one Collection of Object Representations 525 or a stream of Collections of Object Representations 525 as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to Fig. 28, an embodiment of determining anticipatory Instruction Sets 526 using Graph 530b is illustrated. Graph 530b may include knowledge (i.e. interconnected Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527, etc.) of how Device 98 operated in circumstances including objects with various properties. In some aspects, determining anticipatory Instruction Sets 526 using Graph 530b may include selecting a path of Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof through Graph 530b. Individual and/or collective Similarity Comparisons 125 can be used to determine substantial similarity of the individually and/or collectively compared Collections of Object Representations 525 or portions thereof. Substantial similarity may be used primarily for selecting a path through Graph 530b, whereas, weight of any Connection 853 may be used secondarily or not at all.

For example, Decision-making Unit 540 can perform Similarity Comparisons 125 of Collections of Object Representations 525a1-525an or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from one or more Knowledge Cells 800 in Graph 530b. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800ua may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object Representations 525b1-525bn or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from one or more Knowledge Cells 800 in Graph 530b interconnected with Knowledge Cell 800ua by outgoing Connections 853. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800ub may be found collectively substantially similar with highest similarity, thus, Decision-making Unit 540 may follow Connection 853u1 disregarding its less than highest weight. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object Representations 525c1-525cn or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from one or more Knowledge Cells 800 in Graph 530b interconnected with Knowledge Cell 800ub by outgoing Connections 853. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800uc may be found collectively substantially similar with highest similarity, thus, Decision-making Unit 540 may follow Connection 853u2 disregarding its less than highest weight. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-

making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Since Connection 853u3 is the only connection from Knowledge Cell 800uc, Decision-making Unit 540 may follow Connection 853u3 and perform Similarity Comparisons 125 of Collections of Object Representations 525d1-525dn or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cell 800ud in Graph 530b. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800ud may be found collectively substantially similar. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object Representations 525e1-525en or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from one or more Knowledge Cells 800 in Graph 530b interconnected with Knowledge Cell 800ud by outgoing Connections 853. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800ue may be found collectively substantially similar with highest similarity, thus, Decision-making Unit 540 may follow Connection 853u4. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can implement similar logic or process for any additional Collections of Object Representations 525 from Object Processing Unit 93, and so on.

The foregoing exemplary embodiment provides an example of utilizing a combination of collective Similarity Comparisons 125, individual Similarity Comparisons 125, Connections 853, and/or other elements or techniques. It should be understood that any of these elements and/or techniques can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof in a path through Graph 530b would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info 527 in Similarity Comparisons 125, traversing of Knowledge Cells 800 or other elements, using history of Collections of Object Representations 525 or Knowledge Cells 800 in collective Similarity Comparisons 125, using various arrangements of Collections of Object Representations 525 and/or other elements in a Knowledge Cell 800, and/or others can similarly be utilized in this example. These elements and/or techniques can similarly be utilized in Neural Network 530a, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, and/or other data structures or arrangements. In some aspects, instead of anticipating Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially matching streams of Collections of Object Representations 525. In other aspects, any time that substantial similarity or other similarity threshold is not achieved in compared Collections of Object Representations 525 or portions thereof of any of the Knowledge Cells 800, Decision-making Unit 540 can decide to look for a substantially or otherwise similar Collections of Object Representations 525 or portions thereof in Knowledge Cells 800 elsewhere in Graph 530b even if such Knowledge Cell 800 may be unconnected with a prior Knowledge Cell 800. It should be

noted that any of Collections of Object Representations 525a1-525an, Collections of Object Representations 525b1-525bn, Collections of Object Representations 525c1-525cn, Collections of Object Representations 525d1-525dn, Collections of Object Representations 525e1-525en, etc. may include one Collection of Object Representations 525 or a stream of Collections of Object Representations 525. It should also be noted that any Knowledge Cell 800 may include one Collection of Object Representations 525 or a stream of Collections of Object Representations 525 as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to Fig. 29, an embodiment of determining anticipatory Instruction Sets 526 using Collection of Sequences 530c is illustrated. Collection of Sequences 530c may include knowledge (i.e. sequences of Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527, etc.) of how Device 98 operated in circumstances including objects with various properties. In some aspects, determining anticipatory Instruction Sets 526 for autonomous Device 98 operation using Collection of Sequences 530c may include selecting a Sequence 533 of Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof from Collection of Sequences 530c. Individual and/or collective Similarity Comparisons 125 can be used to determine substantial similarity of the individually and/or collectively compared Collections of Object Representations 525 or portions thereof.

For example, Decision-making Unit 540 can perform Similarity Comparisons 125 of Collections of Object Representations 525a1-525an or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in one or more Sequences 533 of Collection of Sequences 530c. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800ca in Sequence 533wc may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object Representations 525a1-525an and 525b1-525bn or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Sequences 533 of Collection of Sequences 530c. Collections of Object Representations 525 or portions thereof from Knowledge Cells 800ca-800cb in Sequence 533wc may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object Representations 525a1-525an, 525b1-525bn, and 525c1-525cn or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Sequences 533 of Collection of Sequences 530c. Collections of Object Representations 525 or portions thereof from Knowledge Cells 800da-800dc in Sequence 533wd may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-

making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object Representations 525a1-525an, 525b1-525bn, 525c1-525cn, and 525d1-525dn or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Sequences 533 of Collection of Sequences 530c. Collections of Object Representations 525 or portions thereof from Knowledge Cells 800da-800dd in Sequence 533wd may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object Representations 525a1-525an, 525b1-525bn, 525c1-525cn, 525d1-525dn, and 525e1-525en or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Sequences 533 of Collection of Sequences 530c. Collections of Object Representations 525 or portions thereof from Knowledge Cells 800da-800de in Sequence 533wd may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can implement similar logic or process for any additional Collections of Object Representations 525 from Object Processing Unit 93, and so on.

The foregoing exemplary embodiment provides an example of utilizing a combination of collective Similarity Comparisons 125, individual Similarity Comparisons 125, and/or other elements or techniques. It should be understood that any of these elements and/or techniques can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Sequence 533 of Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info 527 in Similarity Comparisons 125, traversing of Knowledge Cells 800 or other elements, using history of Collections of Object Representations 525 or Knowledge Cells 800 in collective Similarity Comparisons 125, using various arrangements of Collections of Object Representations 525 and/or other elements in a Knowledge Cell 800, and/or others can similarly be utilized in this example. These elements and/or techniques can similarly be utilized in Neural Network 530a, Graph 530b, Collection of Knowledge Cells 530d, and/or other data structures or arrangements. In some aspects, instead of anticipating Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially matching streams of Collections of Object Representations 525. In other aspects, any time that substantial similarity or other similarity threshold is not achieved in compared Collections of Object Representations 525 or portions thereof from any of the Knowledge Cells 800, Decision-making Unit 540 can decide to look for a substantially or otherwise similar Collections of Object Representations 525 or portions thereof in Knowledge Cells 800 elsewhere in Collection of Sequences 530c such as in different Sequences 533. It should be noted that any of Collections of Object Representations 525a1-525an, Collections of

Object Representations 525b1-525bn, Collections of Object Representations 525c1-525cn, Collections of Object Representations 525d1-525dn, Collections of Object Representations 525e1-525en, etc. may include one Collection of Object Representations 525 or a stream of Collections of Object Representations 525. It should also be noted that any Knowledge Cell 800 may include one Collection of Object Representations 525 or a stream of Collections of Object Representations 525 as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring now to Modification Interface 130. Modification Interface 130 comprises the functionality for modifying execution and/or functionality of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. Modification Interface 130 comprises the functionality for modifying execution and/or functionality of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element at runtime. Modification Interface 130 comprises the functionality for modifying execution and/or functionality of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element based on anticipatory Instruction Sets 526. In one example, Modification Interface 130 comprises the functionality to access, modify, and/or perform other manipulations on runtime engine/environment, virtual machine, operating system, compiler, just-in-time (JIT) compiler, interpreter, translator, execution stack, file, object, data structure, and/or other computing system elements. In another example, Modification Interface 130 comprises the functionality to access, modify, and/or perform other manipulations on memory, storage, bus, interfaces, and/or other computing system elements. In a further example, Modification Interface 130 comprises the functionality to access, modify, and/or perform other manipulations on Processor 11 registers and/or other Processor 11 elements. In a further example, Modification Interface 130 comprises the functionality to access, modify, and/or perform other manipulations on inputs and/or outputs of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. In a further example, Modification Interface 130 comprises the functionality to access, create, delete, modify, and/or perform other manipulations on functions, methods, procedures, routines, subroutines, and/or other elements of Application Program 18. In a further example, Modification Interface 130 comprises the functionality to access, create, delete, modify, and/or perform other manipulations on source code, bytecode, compiled, interpreted, or otherwise translated code, machine code, and/or other code. In a further example, Modification Interface 130 comprises the functionality to access, create, delete, modify, and/or perform other manipulations on values, variables, parameters, and/or other data or information. Modification Interface 130 comprises any features, functionalities, and embodiments of Acquisition Interface 120, and vice versa. Modification Interface 130 also comprises other disclosed functionalities.

Modification Interface 130 can employ various techniques for modifying execution and/or functionality of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. In some aspects, some of the previously described techniques and/or tools can be utilized. Code instrumentation, for instance, may involve inserting additional code, overwriting or rewriting existing code, and/or branching to a separate segment of code in Application Program 18 as previously described. For example, instrumented code may include the following:

```
Device1.moveLeft(23);
modifyApplication();
```

In the above sample code, instrumented call to Modification Interface's 130 function (i.e. modifyApplication(), etc.) can be placed after a function (i.e. Device1.moveLeft(23), etc.) of Application Program 18. Similar call to an

application modifying function can be placed after or before some or all functions/routines/subroutines, some or all lines of code, some or all statements, some or all instructions or instruction sets, some or all basic blocks, and/or some or all other code segments of Application Program 18. One or more application modifying function calls can be placed anywhere in Application Program's 18 code and can be executed at any points in Application Program's 18 execution. The application modifying function (i.e. modifyApplication(), etc.) may include Artificial Intelligence Unit 110-determined anticipatory Instruction Sets 526 that can modify execution and/or functionality of Application Program 18. In some embodiments, the previously described obtaining Application Program's 18 instruction sets, data, and/or other information as well as modifying execution and/or functionality of Application Program 18 can be implemented in a single function that performs both tasks (i.e. traceAndModifyApplication(), etc.).

In some embodiments, various computing systems and/or platforms may provide native tools for modifying execution and/or functionality of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. Independent vendors may provide tools with similar functionalities that can be utilized across different platforms. These tools enable a wide range of techniques or capabilities such as instrumentation, self-modifying code capabilities, dynamic code capabilities, branching, code rewriting, code overwriting, hot swapping, accessing and/or modifying objects or data structures, accessing and/or modifying functions/routines/subroutines, accessing and/or modifying variable or parameter values, accessing and/or modifying processor registers, accessing and/or modifying inputs and/or outputs, providing runtime memory access, and/or other capabilities. One of ordinary skill in art will understand that, while all possible variations of the techniques for modifying execution and/or functionality of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element are too voluminous to describe, these techniques are within the scope of this disclosure.

In one example, modifying execution and/or functionality of Application Program 18 can be implemented through utilizing metaprogramming techniques, which include applications that can self-modify or that can create, modify, and/or manipulate other applications. Self-modifying code, dynamic code, reflection, and/or other techniques can be used to facilitate metaprogramming. In some aspects, metaprogramming is facilitated through a programming language's ability to access and manipulate the internals of the runtime engine directly or via an API. In other aspects, metaprogramming is facilitated through dynamic execution of expressions (i.e. anticipatory Instruction Sets 526, etc.) that can be created and/or executed at runtime. In yet other aspects, metaprogramming is facilitated through application modification tools, which can perform modifications on an application regardless of whether the application's programming language enables metaprogramming capabilities. Some operating systems may protect an application loaded into memory by restricting access to the loaded application. This protection mechanism can be circumvented by utilizing operating system's, processor's, and/or other low level features or commands to unprotect the loaded application. For example, a self-modifying application may modify the in-memory image of itself. To do so, the application can obtain the in-memory address of its code. The application may then change the operating system's or platform's protection on this memory range allowing it to modify the code (i.e. insert anticipatory Instruction Sets 526, etc.). In addition to a self-modifying application, one application can utilize similar technique to modify another application. Linux mprotect command or similar commends of other operating systems can be used to change protection (i.e. unprotect, etc.) for a region of memory, for example. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program 18 can be

implemented through native capabilities of dynamic, interpreted, and/or scripting programming languages and/or platforms. Most of these languages and/or platforms can perform functionalities at runtime that static programming languages may perform during compilation. Dynamic, interpreted, and/or scripting languages provide native functionalities such as self-modification of code, dynamic code, extending the application, adding new code, extending objects and definitions, and/or other functionalities that can modify an application's execution and/or functionality at runtime. Examples of dynamic, interpreted, and/or scripting languages include Lisp, Perl, PHP, JavaScript, Ruby, Python, Smalltalk, Tcl, VBScript, and/or others. Similar functionalities can also be provided in languages such as Java, C, and/or others using reflection. Reflection includes the ability of an application to examine and modify the structure and behavior of the application at runtime. For example, JavaScript can modify its own code as it runs by utilizing Function object constructor as follows:

```
myFunc=new Function(arg1, arg2, argN, functionBody);
```

The sample code above causes a new function object to be created with the specified arguments and body. The body and/or arguments of the new function object may include new instruction sets (i.e. anticipatory Instruction Sets 526, etc.). The new function can be invoked as any other function in the original code. In another example, JavaScript can utilize eval method that accepts a string of JavaScript statements (i.e. anticipatory Instruction Sets 526, etc.) and execute them as if they were within the original code. An example of how eval method can be used to modify an application includes the following JavaScript code:

```
anticipatoryInstr = 'Device1.moveForward(27);';
if (anticipatoryInstr != "" && anticipatoryInstr != null)
{
        eval(anticipatoryInstr);
}
```

In the sample code above, Artificial Intelligence Unit 110 may generate anticipatory Instruction Set 526 (i.e. 'Device1.moveForward(27)' for moving a Device1 forward 27 units, etc.) and save it in anticipatoryInstr variable, which eval method can then execute. Lisp is another example of dynamic, interpreted, and/or scripting language that includes similar capabilities as previously described JavaScript. For example, Lisp's compile command can create a function at runtime, eval command may parse and evaluate an expression at runtime, and exec command may execute a given instruction set (i.e. string, etc.) at runtime. In another example, dynamic as well as some non-dynamic languages may provide macros, which combine code introspection and/or eval capabilities. In some aspects, macros can access inner workings of the compiler, interpreter, virtual machine, runtime environment/engine, and/or other components of the computing platform enabling the definition of language-like constructs and/or generation of a complete program or sections thereof. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program 18 can be implemented through dynamic code, dynamic class loading, reflection, and/or other native functionalities of a programming language or platform. In static applications or static programming, a class can be defined and/or loaded at compile time. Conversely, in dynamic applications or dynamic programming, a class can be loaded into a running environment at runtime. For example, Java Runtime Environment (JRE) may not require that all classes be loaded at compile time and class loading can occur when a class is first referenced at runtime. Dynamic class

loading enables inclusion or injection of on-demand code and/or functionalities at runtime. System provided or custom class loaders may enable loading of classes into the running environment. Custom class loaders can be created to enable custom functionalities such as, for example, specifying a remote location from which a class can be loaded. In addition to dynamic loading of a pre-defined class, a class can also be created at runtime. In some aspects, a class source code can be created at runtime. A compiler such as javac, com.sun.tools.javac.Main, javax.tools, javax.tools.JavaCompiler, and/or other packages can then be utilized to compile the source code. Javac, com.sun.tools.javac.Main, javax.tools, javax.tools.JavaCompiler, and/or other packages may include an interface to invoke Java compiler from within a running application. A Java compiler may accept source code in a file, string, object (i.e. Java String, StringBuffer, CharSequence, etc.) and/or other source, and may generate Java bytecode (i.e. class file, etc.). Once compiled, a class loader can then load the compiled class into the running environment. In other aspects, a tool such as Javaassist (i.e. Java programming assistant) can be utilized to enable an application to create or modify a class at runtime. Javassist may include a Java library that provides functionalities to create and/or manipulate Java bytecode of an application as well as reflection capabilities. Javassist may provide source-level and bytecode-level APIs. Using the source-level API, a class can be created and/or modified using only source code, which Javassist may compile seamlessly on the fly. Javassist source-level API can therefore be used without knowledge of Java bytecode specification. Bytecode-level API enables creating and/or editing a class bytecode directly. In yet other aspects, similar functionalities to the aforementioned ones may be provided in tools such as Apache Commons BCEL (Byte Code Engineering Library), ObjectWeb ASM, CGLIB (Byte Code Generation Library), and/or others. Once a dynamic code or class is created and loaded, reflection in high-level programming languages such as Java and/or others can be used to manipulate or change the runtime behavior of an application. Examples of reflective programming languages and/or platforms include Java, JavaScript, Smalltalk, Lisp, Python, .NET Common Language Runtime (CLR), Tcl, Ruby, Perl, PHP, Scheme, PL/SQL, and/or others. Reflection can be used in an application to access, examine, modify, and/or manipulate a loaded class and/or its elements. Reflection in Java can be implemented by utilizing a reflection API such as java.lang.Reflect package. The reflection API provides functionalities such as, for example, loading or reloading a class, instantiating a new instance of a class, determining class and instance methods, invoking class and instance methods, accessing and manipulating a class, fields, methods and constructors, determining the modifiers for fields, methods, classes, and interfaces, and/or other functionalities. The above described dynamic code, dynamic class loading, reflection, and/or other functionalities are similarly provided in the .NET platform through its tools such as, for example, System.CodeDom.Compiler namespace, System.Reflection.Emit namespace, and/or other native or other .NET tools. Other platforms in addition to Java and .NET may provide similar tools and/or functionalities. In some designs, dynamic code, dynamic class loading, reflection, and/or other functionalities can be used to facilitate modification of an application by inserting or injecting instruction sets (i.e. anticipatory Instruction Sets 526, etc.) into a running application. For example, an existing or dynamically created class comprising DCADO Unit 100 functionalities can be loaded into a running application through manual, automatic, or dynamic instrumentation. Once the class is created and loaded, an instance of DCADO Unit 100 class may be constructed. The instance of DCADO Unit 100 can then take or exert control of the application and/or implement alternate instruction sets (i.e. anticipatory Instruction Sets 526, etc.) at any point in the application's execution. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program 18 can be implemented through independent tools that can be utilized across different platforms. Such tools provide instrumentation and/or other capabilities on more than one platform or computing system and may facilitate application modification or insertion of instruction sets (i.e. anticipatory Instruction Sets 526, etc.). Examples of these tools include Pin, DynamoRIO, DynInst, Kprobes, KernInst, OpenPAT, DTrace, SystemTap, and/or others. In some aspects, Pin and/or any of its elements, methods, and/or techniques can be utilized for dynamic instrumentation. Pin can perform instrumentation by taking control of an application after it loads into memory. Pin may insert itself into the address space of an executing application enabling it to take control. Pin JIT compiler can then compile and implement alternate code (i.e. anticipatory Instruction Sets 526, etc.). Pin provides an extensive API for instrumentation at several abstraction levels. Pin supports two modes of instrumentation, JIT mode and probe mode. JIT mode uses a just-in-time compiler to insert instrumentation and recompile program code while probe mode uses code trampolines for instrumentation. Pin was designed for architecture and operating system independence. In other aspects, KernInst and/or any of its elements, methods, and/or techniques can be utilized for dynamic instrumentation. KernInst includes an instrumentation framework designed for dynamically inserting code into a running kernel of an operating system. KernInst implements probe-based dynamic instrumentation where code can be inserted, changed, and/or removed at will. Kerninst API enables client tools to construct their own tools for dynamic kernel instrumentation to suit variety of purposes such as insertion of alternate instruction sets (i.e. anticipatory Instruction Sets 526, etc.). Client tools can communicate with KernInst over a network (i.e. internet, wireless network, LAW, WAN, etc). Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program 18 can be implemented through utilizing operating system's native tools or capabilities such as Unix ptrace command. Ptrace includes a system call that may enable one process to control another allowing the controller to inspect and manipulate the internal state of its target. Ptrace can be used to modify a running application such as modify an application with alternate instruction sets (i.e. anticipatory Instruction Sets 526, etc.). By attaching to an application using the ptrace call, the controlling application can gain extensive control over the operation of its target. This may include manipulation of its instruction sets, execution path, file descriptors, memory, registers, and/or other components. Ptrace can single-step through the target's code, observe and intercept system calls and their results, manipulate the target's signal handlers, receive and send signals on the target's behalf, and/or perform other operations within the target application. Ptrace's ability to write into the target application's memory space enables the controller to modify the running code of the target application. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program 18 can be implemented through utilizing just-in-time (JIT) compiling. JIT compilation (also known as dynamic translation, dynamic compilation, etc.) includes compilation performed during an application's execution (i.e. runtime, etc.). A code can be compiled when it is about to be executed, and it may be cached and reused later without the need for additional compilation. In some aspects, a JIT compiler can convert source code or byte code into machine code. In other aspects, a JIT compiler can convert source code into byte code. JIT compiling may be performed directly in memory. For example, JIT compiler can output machine code directly into memory and immediately execute it.

Platforms such as Java, .NET, and/or others may implement JIT compilation as their native functionality. Platform independent tools for custom system design may include JIT compilation functionalities as well. In some aspects, JIT compilation includes redirecting application's execution to a JIT compiler from a specific entry point. For example, Pin can insert its JIT compiler into the address space of an application. Once execution is redirected to it, JIT compiler may receive alternate instruction sets (i.e. anticipatory Instruction Sets 526, etc.) immediately before their compilation. The JIT compiled instruction sets can be stored in memory or another repository from where they may be retrieved and executed. Alternatively, for example, JIT compiler can create a copy of the original application code or a segment thereof, and insert alternate code (i.e. anticipatory Instruction Sets 526, etc.) before compiling the modified code copy. In some aspects, JIT compiler may include a specialized memory such as fast cache memory dedicated to JIT compiler functionalities from which the modified code can be fetched rapidly. JIT compilation and/or any compilation in general may include compilation, interpretation, or other translation into machine code, bytecode, and/or other formats or types of code. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program 18 can be implemented through dynamic recompilation. Dynamic recompilation includes recompiling an application or part thereof during execution. An application can be modified with alternate features or instruction sets that may take effect after recompilation. Dynamic recompilation may be practical in various types of applications including object oriented, event driven, forms based, and/or other applications. In a typical windows-based application, most of the action after initial startup occurs in response to user or system events such as moving the mouse, selecting a menu option, typing text, running a scheduled task, making a network connection, and/or other events when an event handler is called to perform an operation appropriate for the event. Generally, when no events are being generated, the application is idle. For example, when an event occurs and an appropriate event handler is called, instrumentation can be implemented in the application's source code to insert alternate instruction sets (i.e. anticipatory Instruction Sets 526, etc.) at which point the modified source code can be recompiled and/or executed. In some aspects, the state of the application can be saved before recompiling its modified source code so that the application may continue from its prior state. Saving the application's state can be achieved by saving its variables, data structures, objects, location of its current instruction, and/or other necessary information in environmental variables, memory, or other repositories where they can be accessed once the application is recompiled. In other aspects, application's variables, data structures, objects, address of its current instruction, and/or other necessary information can be saved in a repository such as file, database, or other repository accessible to the application after recompilation of its source code. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program 18 can be implemented through modifying or redirecting Application Program's 18 execution path. Generally, an application can be loaded into memory and the flow of execution proceeds from one instruction set to the next until the end of the application. An application may include a branching mechanism that can be driven by keyboard or other input devices, system events, and/or other computing system components or events that may impact the execution path. The execution path can also be altered by an external application through acquiring control of execution and/or redirecting execution to a function, routine/subroutine, or an alternate code segment at any point in the application's

execution. A branch, jump, or other mechanism can be utilized to implement the redirected execution. For example, a jump instruction can be inserted at a specific point in an application's execution to redirect execution to an alternate code segment. A jump instruction set may include, for example, an unconditional branch, which always results in branching, or a conditional branch, which may or may not result in branching depending on a condition. When executing an application, a computer may fetch and execute instruction sets in sequence until it encounters a branch instruction set. If the instruction set is an unconditional branch, or it is conditional and the condition is satisfied, the computer may fetch its next instruction set from a different instruction set sequence or code segment as specified by the branch instruction set. After the execution of the alternate code segment, control may be redirected back to the original jump point or to another point in the application. For example, modifying an application can be implemented by redirecting execution of an application to alternate instruction sets (i.e. anticipatory Instruction Sets 526, etc.). Alternate instruction sets can be pre-compiled, pre-interpreted, or otherwise pre-translated and ready for execution. Alternate instruction sets can also be JIT compiled, JIT interpreted, or otherwise JIT translated before execution. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program 18 can be implemented through assembly language. Assembly language instructions may be directly related with the architecture's machine instructions as previously described. Assembly language can, therefore, be a powerful tool for implementing direct hardware (i.e. processor registers, memory, etc.) access and manipulations as well as access and manipulations of specialized processor features or instructions. Assembly language can also be a powerful tool for implementing low-level embedded systems, real-time systems, interrupt handlers, self or dynamically modifying code, and/or other applications. Specifically, for instance, self or dynamically modifying code that can be used to facilitate modifying of an application can be seamlessly implemented using assembly language. For example, using assembly language, instruction sets can be dynamically created and loaded into memory similar to the ones that a compiler may generate. Furthermore, using assembly language, memory space of a loaded application can be accessed to modify (including rewrite, overwrite, etc.) original instruction sets or to insert jumps or branches to alternate code elsewhere in memory. Some operating systems may implement protection from changes to applications loaded into memory. Operating system's, processor's, or other low level features or commands can be used to unprotect the protected locations in memory before the change as previously described. Alternatively, a pointer that may reside in a memory location where it could be readily altered can be utilized where the pointer may reference alternate code. In one example, assembly language can be utilized to write alternate code (i.e. anticipatory Instruction Sets 526, etc.) into a location in memory outside a running application's memory space. Assembly language can then be utilized to redirect the application's execution to the alternate code by inserting a jump or branch into the application's in-memory code, by redirecting program counter, or by other technique. In another example, assembly language can be utilized to overwrite or rewrite the entire or part of an application's in-memory code with alternate code. In some aspects, high-level programming languages can call an external assembly language program to facilitate application modification as previously described. In yet other aspects, relatively low-level programming languages such as C may allow embedding assembly language directly in their source code such as, for example, using asm keyword of C. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program 18 can be implemented through binary rewriting. Binary rewriting tools and/or techniques may modify an application's executable. In some aspects, modification can be minor such as in the case of optimization where the original executable's functionality is kept. In other aspects, modification may change the application's functionality such as by inserting alternate code (i.e. anticipatory Instruction Sets 526, etc.). Examples of binary rewriting tools include SecondWrite, ATOM, DynamoRIO, Purify, Pin, EEL, DynInst, PLTO, and/or others. Binary rewriting may include disassembly, analysis, and/or modification of target application. Since binary rewriting works directly on machine code executable, it is independent of source language, compiler, virtual machine (if one is utilized), and/or other higher level abstraction layers. Also, binary rewriting tools can perform application modifications without access to original source code. Binary rewriting tools include static rewriters, dynamic rewriters, minimally-invasive rewriters, and/or others. Static binary rewriters can modify an executable when the executable is not in use (i.e. not running). The rewritten executable may then be executed including any new or modified functionality. Dynamic binary rewriters can modify an executable during its execution, thereby enabling modification of an application's functionality at runtime. In some aspects, dynamic rewriters can be used for instrumentation or selective modifications such as insertion of alternate code (i.e. anticipatory Instruction Sets 526, etc.), and/or for other runtime transformations or modifications. For example, some dynamic rewriters can be configured to intercept an application's execution at indirect control transfers and insert instrumentation or other application modifying code. Minimally-invasive rewriters may keep the original machine code to the greatest extent possible. They support limited modifications such as insertion of jumps into and out of instrumented code. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

Referring to Fig. 30, in a further example, modifying execution and/or functionality of Processor 11 can be implemented through modification of processor registers, memory, or other computing system components. In some aspects, modifying execution and/or functionality of Processor 11 can be implemented by redirecting Processor's 11 execution to alternate instruction sets (i.e. anticipatory Instruction Sets 526, etc.). In one example, Program Counter 211 may hold or point to a memory address of the next instruction set that will be executed by Processor 11. Artificial Intelligence Unit 110 may generate anticipatory Instruction Sets 526 and store them in Memory 12 as previously described. Modification Interface 130 may then change Program Counter 211 to point to the location in Memory 12 where anticipatory Instruction Sets 526 are stored. The anticipatory Instruction Sets 526 can then be fetched from the location in Memory 12 pointed to by the modified Program Counter 211 and loaded into Instruction Register 212 for decoding and execution. Once anticipatory Instruction Sets 526 are executed, Modification Interface 130 may change Program Counter 211 to point to the last instruction set before the redirection or to any other instruction set. In other aspects, anticipatory Instruction Sets 526 can be loaded directly into Instruction Register 212. As previously described, examples of other processor or computing system components that can be used during an instruction cycle include memory address register (MAR), memory data register (MDR), data registers, address registers, general purpose registers (GPRs), conditional registers, floating point registers (FPRs), constant registers, special purpose registers, machine-specific registers, Register Array 214, Arithmetic Logic Unit 215, control unit, and/or other circuits or components. Any of the aforementioned processor registers, memory, or other computing system components can be accessed and/or modified to facilitate the disclosed functionalities. In some embodiments, processor interrupt may be issued to facilitate such access and/or modification. In some

designs, modifying execution and/or functionality of Processor 11 can be implemented in a program, combination of programs and hardware, or purely hardware system. Dedicated hardware may be built to perform modifying execution and/or functionality of Processor 11 with marginal or no impact to computing overhead. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

Referring to Figs. 31A-31B, in a further example, modifying execution and/or functionality of Logic Circuit 250 can be implemented through modification of inputs and/or outputs of Logic Circuit 250. While Processor 11 includes any type of logic circuit, Logic Circuit 250 is described separately herein to offer additional detail on its functioning. Logic Circuit 250 comprises the functionality for performing logic operations using the circuit's inputs and producing outputs based on the logic operations performed as previously described. In one example, Logic Circuit 250 may perform some logic operations using four input values and produce two output values. Modifying execution and/or functionality of Logic Circuit 250 can be implemented by replacing its input values with anticipatory input values (i.e. anticipatory Instruction Sets 526, etc.). Artificial Intelligence Unit 110 may generate anticipatory input values as previously described. Modification Interface 130 can then transmit the anticipatory input values to Logic Circuit 250 through the four hardwired connections as shown in Fig. 31A. Modification Interface 130 may use Switches 251 to prevent delivery of any input values that may be sent to Logic Circuit 250 from its usual input source. As such, DCADO Unit 100 may cause Logic Circuit 250 to perform its logic operations using the four anticipatory input values, thereby implementing autonomous Device 98 operation. In another example, Logic Circuit 250 may perform some logic operations using four input values and produce two output values. Modifying execution and/or functionality of Logic Circuit 250 can be implemented by replacing its output values with anticipatory output values (i.e. anticipatory Instruction Sets 526, etc.). Artificial Intelligence Unit 110 may generate anticipatory output values (i.e. anticipatory Instruction Sets 526, etc.) as previously described. Modification Interface 130 can then transmit the anticipatory output values through the two hardwired connections as shown in Fig. 31B. Modification Interface 130 may use Switches 251 to prevent delivery of any output values that may be sent by Logic Circuit 250. As such, DCADO Unit 100 may bypass Logic Circuit 250 and transmit the two anticipatory output values to downstream elements, thereby implementing autonomous Device 98 operation. In a further example, instead of or in addition to modifying input and/or output values of Logic Circuit 250, the execution and/or functionality of Logic Circuit 250 may be modified by modifying values or signals in one or more Logic Circuit's 250 internal components such as registers, memories, buses, and/or others (i.e. similar to the previously described modifying of Processor 11 components, etc.). In some designs, modifying execution and/or functionality of Logic Circuit 250 can be implemented in a program, combination of programs and hardware, or purely hardware system. Dedicated hardware may be built to perform modifying execution and/or functionality of Logic Circuit 250 with marginal or no impact to computing overhead. Any of the elements and/or techniques for modifying execution and/or functionality of Logic Circuit 250 can similarly be implemented with Processor 11 and/or other processing elements.

In some embodiments, DCADO Unit 100 may directly modify the functionality of an actuator (previously described, not shown). For example, Logic Circuit 250 or other processing element may control an actuator that enables Device 98 to perform mechanical, physical, and/or other operations. An actuator may receive one or more input values or control signals from Logic Circuit 250 or other processing element directing the actuator to perform specific operations. Modifying functionality of an actuator can be implemented by replacing its input values with anticipatory input values (i.e. anticipatory Instruction Sets 526, etc.) as previously described with respect to replacing

input values of Logic Circuit 250. Specifically, for instance, Artificial Intelligence Unit 110 may generate anticipatory input values as previously described. Modification Interface 130 can then transmit the anticipatory input values to the actuator. Modification Interface 130 may use Switches 251 to prevent delivery of any input values that may be sent to the actuator from its usual input source. As such, DCADO Unit 100 may cause the actuator to perform its operations using the anticipatory input values, thereby implementing autonomous Device 98 operation.

One of ordinary skill in art will recognize that Figs. 31A-31B depict one of many implementations of Logic Circuit 250 and that any number of input and/or output values can be utilized in alternate implementations. One of ordinary skill in art will also recognize that Logic Circuit 250 may include any number and/or combination of logic components to implement any logic operations.

Other additional techniques or elements can be utilized as needed for modifying execution and/or functionality of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing elements, or some of the disclosed techniques or elements can be excluded, or a combination thereof can be utilized in alternate embodiments.

Referring to Fig. 32, the illustration shows an embodiment of a method 9100 for learning and/or using a device's circumstances for autonomous device operation. In some aspects, the method can be used on a computing enabled device or system to enable learning of a device's operation in circumstances including objects with various properties and enable autonomous device operation in similar circumstances. Method 9100 may include any action or operation of any of the disclosed methods such as method 9200, 9300, 9400, 9500, 9600, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method 9100.

At step 9105, a first collection of object representations is received. A collection of object representations (i.e. Collection of Object Representations 525, etc.) may include one or more object representations (i.e. Object Representations 625, etc.), object properties (i.e. Object Properties 630, etc.), and/or other elements or information. An object representation may include an electronic representation of an object (i.e. Object 615, etc.) detected in a device's surrounding. In some aspects, a collection of object representations may include one or more object representations, object properties, and/or other elements or information detected in a device's (i.e. Device's 98, etc.) surrounding at a particular time. A collection of object representations may, therefore, include knowledge (i.e. unit of knowledge, etc.) of a device's circumstances including objects with various properties at a particular time. In some designs, a collection of object representations may include or be associated with a time stamp (not shown), order (not shown), or other time related information. In some embodiments, a collection of object representations may include or be substituted with a stream of collections of object representations, and vice versa. Therefore, the terms collection of object representations and stream of collections of object representations may be used interchangeably herein depending on context. A stream of collections of object representations may include one collection of object representations or a group, sequence, or other plurality of collections of object representations. In some aspects, a stream of collections of object representations may include one or more collections of object representations, and/or other elements or information detected in a device's surrounding over time. A stream of collections of object representations may, therefore, include knowledge (i.e. unit of knowledge, etc.) of a device's circumstances including objects with various properties over time. As circumstances including objects with various properties in a device's surrounding change (i.e. objects and/or their properties change, move, act, transform, etc.) over time, this

change may be captured in a stream of collections of object representations. In some designs, each collection of object representations in a stream may include or be associated with the aforementioned time stamp, order, or other time related information. Examples of objects include biological objects (i.e. persons, animals, vegetation, etc.), nature objects (i.e. rocks, bodies of water, etc.), manmade objects (i.e. buildings, streets, ground/aerial/aquatic vehicles, etc.), and/or others. In some aspects, any part of an object may be detected as an object itself. For instance, instead of or in addition to detecting a vehicle as an object, a wheel and/or other parts of the vehicle may be detected as objects. In general, an object may include any object or part thereof that can be detected. Examples of object properties include existence of an object, type of an object (i.e. person, cat, vehicle, building, street, tree, rock, etc.), identity of an object (i.e. name, identifier, etc.), distance of an object, bearing/angle of an object, location of an object (i.e. distance and bearing/angle from a known point, object coordinates, etc.), shape/size of an object (i.e. height, width, depth, computer model, point cloud, etc.), activity of an object (i.e. motion, gestures, etc.), and/or other properties of an object. In general, an object property may include any attribute of an object (i.e. existence of an object, type of an object, identity of an object, shape/size of an object, etc.), any relationship of an object with the device, other objects, or the environment (i.e. distance of an object, bearing/angle of an object, friend/foe relationship, etc.), and/or other information related to an object. Objects and/or their properties can be detected by one or more sensors (i.e. Sensors 92, etc.) and/or an object processing unit (i.e. Object Processing Unit 93, etc.). A sensor may obtain or detect information about its environment. As such, one or more sensors can be used to detect objects and/or their properties in a device's surrounding. In some designs, a sensor may be part of a device whose circumstances are being used for DCADO functionalities. In other designs, a sensor may be part of a remote device whose circumstances are being used for DCADO functionalities. Examples of a sensor include a camera (i.e. Camera 92a, etc.), a microphone (i.e. Microphone 92b, etc.), a lidar (i.e. Lidar 92c, etc.), a radar (i.e. Radar 92d, etc.), a sonar (i.e. Sonar 92e, etc.), and/or others. An object processing unit may process output from a sensor to obtain information of interest. As such, an object processing unit can be used to process output from a sensor to detect objects and/or their properties in a device's surrounding. In some aspects, an object processing unit may create or generate a collection of object representations. In other aspects, an object processing unit may create or generate a stream of collections of object representations. An object processing unit may include a picture recognizer (i.e. Picture Recognizer 94a, etc.), a sound recognizer (i.e. Sound Recognizer 94b, etc.), a lidar processing unit (i.e. Lidar Processing Unit 94c, etc.), a radar processing unit (i.e. Radar Processing Unit 94d, etc.), a sonar processing unit (i.e. Sonar Processing Unit 94e, etc.), and/or other elements or functionalities. In general, an object processing unit may include any signal processing element or technique known in art as applicable. In some implementations, an object processing unit and/or any of its elements or functionalities can be included in sensor and/or other elements. Receiving comprises any action or operation by or for a Collection of Object Representations 525, stream of Collections of Object Representations 525, Object Representation 625, Object Property 630, Sensor 92, Camera 92a, Microphone 92b, Lidar 92c, Radar 92d, Sonar 92e, Object Processing Unit 93, Picture Recognizer 94a, Sound Recognizer 94b, Lidar Processing Unit 94c, Radar Processing Unit 94d, Sonar Processing Unit 94e, and/or other disclosed elements.

At step 9110, a first one or more instruction sets for operating a device are received. In some embodiments, an instruction set (i.e. Instruction Set 526, etc.) may be used or executed by a processor (i.e. Processor 11, etc.) in operating a device. In other embodiments, an instruction set may be part of an application program (i.e. Application

Program 18, etc.) used in operating a device. For example, the application can run or execute on one or more processors or other processing elements. In further embodiments, an instruction set may be used or executed by a logic circuit (i.e. Logic Circuit 250, etc.) in operating a device. For example, such instruction set may be or include one or more inputs into or outputs from a logic circuit. In further embodiments, an instruction set may be used by an actuator in operating a device. For example, such instruction set may be or include one or more inputs into an actuator. Operating a device includes performing or causing any operations on/by/with the device. In some designs, an instruction set can be received from a processor, application program, logic circuit, and/or other processing element as the instruction set is being used or executed. In other aspects, an instruction set can be received from a processor, application program, logic circuit, and/or other processing element after the instruction set is used or executed. In further aspects, an instruction set can be received from a processor, application program, logic circuit, and/or other processing element before the instruction set has been used or executed. In further aspects, an instruction set can be received from a running processor, running application program, running logic circuit, and/or other running processing element. As such, an instruction set can be received at runtime. In other designs, an instruction set can be received from an actuator. In further designs, an instruction set can be received from memory (i.e. Memory 12, etc.), hard drive, or any other storage element or repository. In further designs, an instruction set can be received over a network such as Internet, local area network, wireless network, and/or other network. In further designs, an instruction set can be received by an interface (i.e. Acquisition Interface 120, etc.) configured to obtain instruction sets from a processor, application program, logic circuit, actuator, and/or other element. In general, an instruction set can be received by any element of the system. One or more instruction sets may temporally correspond to a collection of object representations. In some aspects, an instruction set that temporally corresponds to a collection of object representations includes an instruction set used or executed at the time of generating the collection of object representations. In other aspects, an instruction set that temporally corresponds to a collection of object representations includes an instruction set used or executed within a certain time period before and/or after generating the collection of object representations. Any time period can be utilized depending on implementation. In further aspects, an instruction set that temporally corresponds to a collection of object representations includes an instruction set used or executed from the time of generating the collection of object representations to the time of generating a next collection of object representations. In further aspects, an instruction set that temporally corresponds to a collection of object representations includes an instruction set used or executed from the time of generating a preceding collection of object representations to the time of generating the collection of object representations. Any other temporal relationship or correspondence between collections of object representations and correlated instruction sets can be implemented. In general, one or more instruction sets that temporally correspond to a collection of object representations enable structuring knowledge of a device's operation at or around the time of generating the collection of object representations. Such functionality enables spontaneous or seamless learning of a device's operation in circumstances including objects with various properties as the device is operated in real life situations. In some embodiments, an instruction set may include one or more commands, keywords, symbols (i.e. parentheses, brackets, commas, semicolons, etc.), instructions, operators (i.e. =, <, >, etc.), variables, values, objects, data structures, functions (i.e. Function1(), FIRST(), MIN(), SQRT(), etc.), parameters, states, signals, inputs, outputs, references thereto, and/or other components. In other embodiments, an instruction set may include source code, bytecode, intermediate code, compiled, interpreted, or otherwise translated code,

runtime code, assembly code, machine code, and/or any other computer code. In further embodiments, an instruction set may include one or more inputs into and/or outputs from a logic circuit. In further embodiments, an instruction set may include one or more inputs into an actuator. Receiving comprises any action or operation by or for an Acquisition Interface 120, Instruction Set 526, and/or other disclosed elements.

At step 9115, the first collection of object representations is correlated with the first one or more instruction sets for operating the device. In some aspects, individual collections of object representations can be correlated with one or more instruction sets. In other aspects, streams of collections of object representations can be correlated with one or more instruction sets. In further aspects, individual collections of object representations or streams of collections of object representations can be correlated with the aforementioned temporally corresponding instruction sets. In further aspects, a collection of object representations or stream of collections of object representations may not be correlated with any instruction sets. Correlating may include structuring or generating a knowledge cell (i.e. Knowledge Cell 800, etc.) and storing one or more collections of object representations correlated with any instruction sets into the knowledge cell. Therefore, a knowledge cell may include any data structure or arrangement that can facilitate such storing. A knowledge cell includes knowledge (i.e. unit of knowledge, etc.) of how a device operated in a circumstance including objects with various properties. In some designs, extra information (i.e. Extra Info 527, etc.) may optionally be used to facilitate enhanced comparisons or decision making in autonomous device operation where applicable. Therefore, any collection of object representations, instruction set, and/or other element may include or be correlated with extra information. Extra information may include any information useful in comparisons or decision making performed in autonomous device operation. Examples of extra information include time information, location information, computed information, contextual information, and/or other information. Correlating may be omitted where learning of a device's operation in circumstances including objects with various properties is not implemented. Correlating comprises any action or operation by or for a Knowledge Structuring Unit 520, Knowledge Cell 800, and/or other disclosed elements.

At step 9120, the first collection of object representations correlated with the first one or more instruction sets for operating the device are stored. A collection of object representations correlated with one or more instruction sets may be part of a stored plurality of collections of object representations correlated with one or more instruction sets. Collections of object representations correlated with any instruction sets can be stored in a memory unit or other repository. The aforementioned knowledge cells comprising collections of object representations correlated with any instruction sets can be used in/as neurons, nodes, vertices, or other elements in any of the data structures or arrangements (i.e. neural networks, graphs, sequences, collection of knowledge cells, etc.) used for storing the knowledge of a device's operation in circumstances including objects with various properties. Knowledge cells may be connected, interrelated, or interlinked into knowledge structures using statistical, artificial intelligence, machine learning, and/or other models or techniques. Such interconnected or interrelated knowledge cells can be used for enabling autonomous device operation. The interconnected or interrelated knowledge cells may be stored or organized into a knowledgebase (i.e. Knowledgebase 530, etc.). In some embodiments, knowledgebase may be or include a neural network (i.e. Neural Network 530a, etc.). In other embodiments, knowledgebase may be or include a graph (i.e. Graph 530b, etc.). In further embodiments, knowledgebase may be or include a collection of sequences (i.e. Collection of Sequences 530c, etc.). In further embodiments, knowledgebase may be or include a sequence (i.e. Sequence 533, etc.). In further embodiments, knowledgebase may be or include a collection of knowledge cells

(i.e. Collection of Knowledge Cells 530d, etc.). In general, knowledgebase may be or include any data structure or arrangement, and/or repository capable of storing the knowledge of a device's operation in circumstances including objects with various properties. Knowledgebase may also include or be substituted with various artificial intelligence methods, systems, and/or models for knowledge structuring, storing, and/or representation such as deep learning, supervised learning, unsupervised learning, neural networks (i.e. convolutional neural network, recurrent neural network, deep neural network, etc.), search-based, logic and/or fuzzy logic-based, optimization-based, tree/graph/other data structure-based, hierarchical, symbolic and/or sub-symbolic, evolutionary, genetic, multi-agent, deterministic, probabilistic, statistical, and/or other methods, systems, and/or models. Storing may be omitted where learning of a device's operation in circumstances including objects with various properties is not implemented. Storing comprises any action or operation by or for a Knowledgebase 530, Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, Knowledge Cell 800, Node 852, Layer 854, Connection 853, Similarity Comparison 125, and/or other disclosed elements.

At step 9125, a new collection of object representations is received. Step 9125 may include any action or operation described in Step 9105 as applicable.

At step 9130, the new collection of object representations is compared with the first collection of object representations. Comparing one collection of object representations with another collection of object representations may include comparing at least a portion of one collection of object representations with at least a portion of the other collection of object representations. In some embodiments, collections of object representations may be compared individually. In some aspects, comparing of individual collections of object representations may include comparing one or more object representations of one collection of object representations with one or more object representations of another collection of object representations. In other aspects, comparing of object representations may include comparing one or more object properties of one object representation with one or more object properties of another object representation. In some designs, one or more object properties in the same category (i.e. Category 635, etc.) can be compared. Comparing may include any techniques for comparing text, numbers, and/or other data. In further aspects, some object representations, object properties, and/or other elements of a collection of object representations can be omitted from comparison depending on implementation. In other embodiments, collections of object representations may be compared collectively as part of streams of collections of object representations. Collective comparing of collections of object representations may include any features, functionalities, and embodiments of the aforementioned individual comparing of collections of object representations. In some aspects, collective comparing of collections of object representations may include comparing one or more collections of object representations of one stream of collections of object representations with one or more collections of object representations of another stream of collections of object representations. In some designs, one or more corresponding (i.e. similarly ordered, temporally related, etc.) collections of object representations from the compared streams of collections of object representations can be compared. In other designs, Dynamic Time Warping (DTW) and/or other techniques can be utilized for comparison and/or aligning temporal sequences (i.e. streams of collections of object representations, etc.) that may vary in time or speed. In further aspects, some collections of object representations can be omitted from comparison depending on implementation. Any combination of the aforementioned and/or other elements or techniques can be utilized in alternate embodiments of the comparing. Comparing may be omitted where anticipating of a device's operation in

circumstances including objects with various properties is not implemented. Comparing comprises any action or operation by or for a Decision-making Unit 540, Similarity Comparison 125, and/or other disclosed elements.

At step 9135, a determination is made that there is at least a partial match between the new collection of object representations and the first collection of object representations. In some embodiments, determining at least a partial match between individually compared collections of object representations includes determining that a similarity between one or more portions of one collection of object representations and one or more portions of another collection of object representations exceeds a similarity threshold. In other embodiments, determining at least a partial match between individually compared collections of object representations includes determining at least a partial match between one or more portions of one collection of object representations and one or more portions of another collection of object representations. In further embodiments, determining at least a partial match between individually compared collections of object representations includes determining substantial similarity between one or more portions of one collection of object representations and one or more portions of another collection of object representations. A portion of a collection of object representations may include an object representation, an object property, and/or other portion or element of the collection of object representations. In further embodiments, determining at least a partial match between individually compared collections of object representations includes determining that the number or percentage of matching or substantially matching object representations of the compared collections of object representations exceeds a threshold number (i.e. 1, 2, 4, 7, 18, etc.) or threshold percentage (i.e. 41%, 62%, 79%, 85%, 93%, etc.). In some aspects, type of object representations, importance of object representations, and/or other elements or techniques relating to object representations can be utilized for determining similarity using object representations. In further aspects, some of the object representations can be omitted in determining similarity using object representations depending on implementation. In further embodiments, determining a match or substantial match between compared object representations includes determining that the number or percentage of matching or substantially matching object properties of the compared object representations exceeds a threshold number (i.e. 1, 2, 3, 6, 11, etc.) or a threshold percentage (i.e. 55%, 61%, 78%, 82%, 99%, etc.). In some aspects, categories of object properties, importance of object properties, and/or other elements or techniques relating to object properties can be utilized for determining similarity using object properties. In further aspects, some of the object properties can be omitted in determining similarity using object properties depending on implementation. In some designs, substantial similarity of individually compared collections of object representations can be achieved when a similarity between one or more portions of one collection of object representations and one or more portions of another collection of object representations exceeds a similarity threshold. In other designs, substantial similarity of individually compared collections of object representations can be achieved when the number or percentage of matching or substantially matching object representations of the compared collections of object representations exceeds a threshold number (i.e. 1, 2, 4, 7, 18, etc.) or threshold percentage (i.e. 41%, 62%, 79%, 85%, 93%, etc.). In further aspects, substantial similarity of compared object representations can be achieved when the number or percentage of matching or substantially matching object properties of the compared object representations exceeds a threshold number (i.e. 1, 2, 3, 6, 11, etc.) or a threshold percentage (i.e. 55%, 61%, 78%, 82%, 99%, etc.). In some embodiments, determining at least a partial match between collectively compared collections of object representations (i.e. streams of collections of object representations, etc.) includes determining that the number or percentage of matching or

substantially matching collections of object representations of the compared streams of collections of object representations exceeds a threshold number (i.e. 1, 2, 4, 9, 33, 138, etc.) or threshold percentage (i.e. 39%, 58%, 77%, 88%, 94%, etc.). In some aspects, importance of collections of object representations, order of collections of object representations, and/or other elements or techniques relating to collections of object representations can be utilized for determining similarity of collectively compared collections of object representations or streams of collections of object representations. In further aspects, some of the collections of object representations can be omitted in determining similarity of collectively compared collections of object representations or streams of collections of object representations depending on implementation. In some designs, a threshold for a number or percentage similarity can be used to determine a match or substantial match between any of the aforementioned elements. Any text, number, and/or other data similarity determination techniques can be used in any of the aforementioned similarity determinations. A partial match of any of the compared elements may include a substantially or otherwise similar match, and vice versa. Therefore, these terms may be used interchangeably herein depending on context. Although, substantial similarity or substantial match is frequently used herein, it should be understood that any level of similarity, however high or low, may be utilized as defined by the rules (i.e. thresholds, etc.) for similarity. Any combination of the aforementioned and/or other elements or techniques can be utilized in alternate embodiments. Determining may be omitted where anticipating of a device's operation in circumstances including objects with various properties is not implemented. Determining comprises any action or operation by or for a Decision-making Unit 540, Similarity Comparison 125, and/or other disclosed elements.

At step 9140, the first one or more instruction sets for operating the device correlated with the first collection of object representations are executed. Executing may be performed in response to the aforementioned determining. Executing may be caused by DCADO Unit 100, Artificial Intelligence Unit 110, Modification Interface 130, and/or other disclosed elements. An instruction set may be executed by a processor (i.e. Processor 11, etc.), application program (i.e. Application Program 18, etc.), logic circuit (i.e. Logic Circuit 250, etc.), and/or other processing element. An instruction set may be executed or acted upon by an actuator. In some aspects, instruction sets (i.e. the one or more instruction sets for operating the device correlated with the first collection of object representations, etc.) anticipated or determined to be used or executed in a device's autonomous operation may be referred to as anticipatory instruction sets, alternate instruction sets, and/or other suitable name or reference. Therefore, these terms can be used interchangeably herein depending on context. Executing may include executing one or more alternate instruction sets (i.e. anticipatory instruction sets, etc.) instead of or prior to an instruction set that would have been executed in a regular course of execution. In some embodiments, executing may include modifying a register or other element of a processor with one or more alternate instruction sets. Executing may also include redirecting a processor to one or more alternate instruction sets. In other embodiments, processor may be or comprises a logic circuit. Executing may further include modifying an element of a logic circuit with one or more alternate instruction sets, redirecting the logic circuit to one or more alternate instruction sets, replacing the inputs into the logic circuit with one or more alternate inputs or instruction sets, and/or replacing the outputs from the logic circuit with one or more alternate outputs or instruction sets. Executing may further include replacing the inputs into an actuator with one or more alternate inputs or instruction sets. In further embodiments, a processor may run an application including instruction sets for operating a device. In some aspects, executing includes executing one or more alternate instruction sets as part of the application. In other aspects, executing includes modifying the

application. In further aspects, executing includes redirecting the application to one or more alternate instruction sets. In further aspects, executing includes modifying one or more instruction sets of the application. In further aspects, executing includes modifying the application's source code, bytecode, intermediate code, compiled code, interpreted code, translated code, runtime code, assembly code, machine code, or other code. In further aspects, executing includes modifying memory, processor register, storage, repository or other element where the application's instruction sets are stored or used. In further aspects, executing includes modifying instruction sets used for operating an object of the application. In further aspects, executing includes modifying an element of a processor, an element of a device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input used in running the application. In further aspects, executing includes modifying the application at source code write time, compile time, interpretation time, translation time, linking time, loading time, runtime, or other time. In further aspects, executing includes modifying one or more of the application's lines of code, statements, instructions, functions, routines, subroutines, basic blocks, or other code segments. In further aspects, executing includes a manual, automatic, dynamic, just in time (JIT), or other instrumentation of the application. In further aspects, executing includes utilizing one or more of a .NET tool, .NET application programming interface (API), Java tool, Java API, operating system tool, independent tool, or other tool for modifying the application. In further aspects, executing includes utilizing a dynamic, interpreted, scripting, or other programming language. In further aspects, executing includes utilizing dynamic code, dynamic class loading, or reflection. In further aspects, executing includes utilizing assembly language. In further aspects, executing includes utilizing metaprogramming, self-modifying code, or an application modification tool. In further aspects, executing includes utilizing just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further aspects, executing includes utilizing dynamic expression creation, dynamic expression execution, dynamic function creation, or dynamic function execution. In further aspects, executing includes adding or inserting additional code into the application's code. In further aspects, executing includes modifying, removing, rewriting, or overwriting the application's code. In further aspects, executing includes branching, redirecting, extending, or hot swapping the application's code. Branching or redirecting an application's code may include inserting a branch, jump, or other means for redirecting the application's execution. Executing comprises any action or operation by or for a Processor 11, Application Program 18, Logic Circuit 250, Modification Interface 130, and/or other disclosed elements.

At step 9145, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations are performed by the device. The one or more operations may be performed in response to the aforementioned executing. In some aspects, an operation includes any operation that can be performed by/with/on a computing enabled device. In other aspects, an operation includes any operation that can be performed by/with/on an actuator. In further aspects, an operation includes any operation that can be performed by/with/on a computer. In general, an operation includes any operation that can be performed by/with/on a device or element thereof. One of ordinary skill in art will recognize that, while all possible variations of operations by/with/on a device are too voluminous to describe and limited only by the device's design and/or user's utilization, all operations are within the scope of this disclosure in various implementations.

Referring to Fig. 33, the illustration shows an embodiment of a method 9200 for learning and/or using a device's circumstances for autonomous device operation. In some aspects, the method can be used on a computing enabled device or system to enable learning of a device's operation in circumstances including objects with various

properties and enable autonomous device operation in similar circumstances. Method 9200 may include any action or operation of any of the disclosed methods such as method 9100, 9300, 9400, 9500, 9600, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method 9200.

At step 9205, a first collection of object representations is received. Step 9205 may include any action or operation described in Step 9105 of method 9100 as applicable.

At step 9210, a first one or more instruction sets for operating a device are received. Step 9210 may include any action or operation described in Step 9110 of method 9100 as applicable.

At step 9215, the first collection of object representations correlated with the first one or more instruction sets for operating the device are learned. Step 9215 may include any action or operation described in Step 9115 and/or Step 9120 of method 9100 as applicable.

At step 9220, a new collection of object representations is received. Step 9220 may include any action or operation described in Step 9125 of method 9100 as applicable.

At step 9225, the first one or more instruction sets for operating the device correlated with the first collection of object representations are anticipated based on at least a partial match between the new collection of object representations and the first collection of object representations. Step 9225 may include any action or operation described in Step 9130 and/or Step 9135 of method 9100 as applicable.

At step 9230, the first one or more instruction sets for operating the device correlated with the first collection of object representations are executed. Step 9230 may include any action or operation described in Step 9140 of method 9100 as applicable.

At step 9235, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations are performed by the device. Step 9235 may include any action or operation described in Step 9145 of method 9100 as applicable.

Referring to Fig. 34, the illustration shows an embodiment of a method 9300 for learning and/or using a device's circumstances for autonomous device operation. In some aspects, the method can be used on a computing enabled device or system to enable learning of a device's operation in circumstances including objects with various properties and enable autonomous device operation in similar circumstances. Method 9300 may include any action or operation of any of the disclosed methods such as method 9100, 9200, 9400, 9500, 9600, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method 9300.

At step 9305, a first stream of collections of object representations is received. Step 9305 may include any action or operation described in Step 9105 of method 9100 as applicable.

At step 9310, a first one or more instruction sets for operating a device are received. Step 9310 may include any action or operation described in Step 9110 of method 9100 as applicable.

At step 9315, the first stream of collections of object representations is correlated with the first one or more instruction sets for operating the device. Step 9315 may include any action or operation described in Step 9115 of method 9100 as applicable.

At step 9320, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device are stored. Step 9320 may include any action or operation described in Step 9120 of method 9100 as applicable.

At step 9325, a new stream of collections of object representations is received. Step 9325 may include any action or operation described in Step 9125 of method 9100 as applicable.

At step 9330, the new stream of collections of object representations is compared with the first stream of collections of object representations. Step 9330 may include any action or operation described in Step 9130 of method 9100 as applicable.

At step 9335, a determination is made that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. Step 9335 may include any action or operation described in Step 9135 of method 9100 as applicable.

At step 9340, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations are executed. Step 9340 may include any action or operation described in Step 9140 of method 9100 as applicable.

At step 9345, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations are performed by the device. Step 9345 may include any action or operation described in Step 9145 of method 9100 as applicable.

Referring to Fig. 35, the illustration shows an embodiment of a method 9400 for learning and/or using a device's circumstances for autonomous device operation. In some aspects, the method can be used on a computing enabled device or system to enable learning of a device's operation in circumstances including objects with various properties and enable autonomous device operation in similar circumstances. Method 9400 may include any action or operation of any of the disclosed methods such as method 9100, 9200, 9300, 9500, 9600, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method 9400.

At step 9405, a first collection of object representations is received. Step 9405 may include any action or operation described in Step 9105 of method 9100 as applicable.

At step 9410, a first one or more inputs are received, wherein the first one or more inputs are also received by a logic circuit, and wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device. Step 9410 may include any action or operation described in Step 9110 of method 9100 as applicable.

At step 9415, the first collection of object representations is correlated with the first one or more inputs. Step 9415 may include any action or operation described in Step 9115 of method 9100 as applicable.

At step 9420, the first collection of object representations correlated with the first one or more inputs are stored. Step 9420 may include any action or operation described in Step 9120 of method 9100 as applicable.

At step 9425, a new collection of object representations is received. Step 9425 may include any action or operation described in Step 9125 of method 9100 as applicable.

At step 9430, the new collection of object representations is compared with the first collection of object representations. Step 9430 may include any action or operation described in Step 9130 of method 9100 as applicable.

At step 9435, a determination is made that there is at least a partial match between the new collection of object representations and the first collection of object representations. Step 9435 may include any action or operation described in Step 9135 of method 9100 as applicable.

At step 9440, the first one or more inputs correlated with the first collection of object representations are received by the logic circuit. Step 9440 may include any action or operation described in Step 9140 of method 9100 as applicable.

At step 9445, one or more operations defined by one or more outputs for operating the device produced by the logic circuit are performed by the device. Step 9445 may include any action or operation described in Step 9145 of method 9100 as applicable.

Referring to Fig. 36, the illustration shows an embodiment of a method 9500 for learning and/or using a device's circumstances for autonomous device operation. In some aspects, the method can be used on a computing enabled device or system to enable learning of a device's operation in circumstances including objects with various properties and enable autonomous device operation in similar circumstances. Method 9500 may include any action or operation of any of the disclosed methods such as method 9100, 9200, 9300, 9400, 9600, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method 9500.

At step 9505, a first collection of object representations is received. Step 9505 may include any action or operation described in Step 9105 of method 9100 as applicable.

At step 9510, a first one or more outputs are received, the first one or more outputs transmitted from a logic circuit, wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device. Step 9510 may include any action or operation described in Step 9110 of method 9100 as applicable.

At step 9515, the first collection of object representations is correlated with the first one or more outputs. Step 9515 may include any action or operation described in Step 9115 of method 9100 as applicable.

At step 9520, the first collection of object representations correlated with the first one or more outputs are stored. Step 9520 may include any action or operation described in Step 9120 of method 9100 as applicable.

At step 9525, a new collection of object representations is received. Step 9525 may include any action or operation described in Step 9125 of method 9100 as applicable.

At step 9530, the new collection of object representations is compared with the first collection of object representations. Step 9530 may include any action or operation described in Step 9130 of method 9100 as applicable.

At step 9535, a determination is made that there is at least a partial match between the new collection of object representations and the first collection of object representations. Step 9535 may include any action or operation described in Step 9135 of method 9100 as applicable.

At step 9540, one or more operations defined by the first one or more outputs correlated with the first collection of object representations are performed by the device. Step 9540 may include any action or operation described in Step 9145 of method 9100 as applicable.

Referring to Fig. 37, the illustration shows an embodiment of a method 9600 for learning and/or using a device's circumstances for autonomous device operation. In some aspects, the method can be used on a computing

enabled device or system to enable learning of a device's operation in circumstances including objects with various properties and enable autonomous device operation in similar circumstances. Method 9600 may include any action or operation of any of the disclosed methods such as method 9100, 9200, 9300, 9400, 9500, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method 9600.

At step 9605, a first collection of object representations is received. Step 9605 may include any action or operation described in Step 9105 of method 9100 as applicable.

At step 9610, a first one or more inputs are received, wherein the first one or more inputs are also received by an actuator, and wherein the actuator is configured to receive inputs and perform motions. Step 9610 may include any action or operation described in Step 9110 of method 9100 as applicable.

At step 9615, the first collection of object representations is correlated with the first one or more inputs. Step 9615 may include any action or operation described in Step 9115 of method 9100 as applicable.

At step 9620, the first collection of object representations correlated with the first one or more inputs are stored. Step 9620 may include any action or operation described in Step 9120 of method 9100 as applicable.

At step 9625, a new collection of object representations is received. Step 9625 may include any action or operation described in Step 9125 of method 9100 as applicable.

At step 9630, the new collection of object representations is compared with the first collection of object representations. Step 9630 may include any action or operation described in Step 9130 of method 9100 as applicable.

At step 9635, a determination is made that there is at least a partial match between the new collection of object representations and the first collection of object representations. Step 9635 may include any action or operation described in Step 9135 of method 9100 as applicable.

At step 9640, the first one or more inputs correlated with the first collection of object representations are received by the actuator. Step 9640 may include any action or operation described in Step 9140 of method 9100 as applicable.

At step 9645, one or more motions defined by the first one or more inputs correlated with the first collection of object representations are performed by the actuator. Step 9645 may include any action or operation described in Step 9145 of method 9100 as applicable.

Referring to Fig. 38, in some exemplary embodiments, Device 98 may be or include Loader 98a. Loader 98a may be operated by User 50 in person or remotely. Loader 98a may include or be coupled to one or more Sensors 92 (i.e. collectively referred to as Sensor 92, etc.) such as Camera 92a, Microphone 92b, Lidar 92c, Radar 92d, Sonar 92e, etc. and/or Object Processing Unit 93 that can detect Objects 615aa-615ad, and/or other elements or information in Loader's 98a surrounding. Object Processing Unit 93 may include Picture Recognizer 94a, Sound Recognizer 94b, Lidar Processing Unit 94c, Radar Processing Unit 94d, Sonar Processing Unit 94e, and/or other elements or functionalities as applicable. Object Processing Unit 93 may create or generate one or more (i.e. stream, etc.) Collections of Object Representations 525 comprising Object Representations 625, Object Properties 630, and/or other elements or information representing Objects 615 detected in Loader's 98a surrounding. Loader 98a may also include or be controlled by Logic Circuit 250 (i.e. microcontroller, etc.), Processor 11 (i.e. including any Application Program 18 running thereon, etc.), and/or other processing element that receives User's 50 (i.e.

operator's, etc.) operating directions and causes desired operations with Loader 98a such as moving, maneuvering, collecting, lifting, unloading, and/or others. User 50 can interact with Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element through inputting operating directions via Human-machine Interface 23 such as one or more steering wheels, levers, pedals, buttons, or other input devices. For instance, responsive to User's 50 manipulating a steering wheel and one or more levers, Logic Circuit 250 or Processor 11 may cause Loader's 98a arm with bucket to collect a load, one or more motors or other actuators to move or maneuver Loader 98a, lifting system (i.e. hydraulic, pneumatic, mechanical, electrical, etc.) to lift a load, and/or arm with bucket to unload a load. Loader 98a may also include or be coupled to DCADO Unit 100. DCADO Unit 100 may be embedded (i.e. integrated, etc.) into or coupled to Loader's 98a Logic Circuit 250, Processor 11, and/or other processing element. DCADO Unit 100 may also be a program embedded (i.e. integrated, etc.) into or interfaced with Application Program 18 running on Processor 11 and/or other processing element. DCADO Unit 100 can obtain Instruction Sets 526 used or executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element. In some aspects, Instruction Sets 526 may include one or more inputs into or outputs from Loader's 98a Logic Circuit 250 (i.e. microcontroller, etc.). In other aspects, Instruction Sets 526 may include one or more instruction sets from Loader's 98a Processor's 11 registers or other components. In further aspects, Instruction Sets 526 may include one or more instruction sets used or executed in Application Program 18. DCADO Unit 100 may also optionally obtain any Extra Info 527 (i.e. time, location, computed, contextual, and/or other information, etc.) related to Loader's 98a operation. As User 50 operates Loader 98a in circumstances including objects with various properties as shown, DCADO Unit 100 may learn Loader's 98a operations in these circumstances by correlating Collections of Object Representations 525 representing Objects 615 detected in Loader's 98a surrounding with one or more Instruction Sets 526 used or executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element. Any Extra Info 527 related to Loader's 98a operation may also optionally be correlated with Collections of Object Representations 525. DCADO Unit 100 may store this knowledge into Knowledgebase 530 (i.e. Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.). In the future, DCADO Unit 110 may compare incoming Collections of Object Representations 525 representing Objects 615 detected in Loader's 98a surrounding with previously learned Collections of Object Representations 525 including optionally using any Extra Info 527 for enhanced decision making. If substantially similar or at least a partial match is found or determined, the Instruction Sets 526 correlated with the previously learned Collections of Object Representations 525 can be autonomously executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element, thereby enabling autonomous operation of Loader 98a in similar circumstances as in previously learned ones. For instance, Loader 98a comprising DCADO Unit 100 may learn User 50-directed collecting, moving, maneuvering, lifting, unloading, and/or other operations in a circumstance that includes Rock 615aa, Pile of Material 615ab, Person 615ac, Truck 615ad, and/or other Objects 615 among which Loader 98a may need to maneuver and/or with which Loader 98a may need to interact. In the future, when a circumstance that includes Objects 615 with similar Object Properties 630 is encountered, Loader 98a may implement collecting, moving, maneuvering, lifting, and/or unloading operations autonomously.

In some embodiments, DCADO Unit 100 may reside on Server 96 accessible over Network 95 as previously described. In such embodiments, any number of Loaders 98a may connect to such remote DCADO Unit

100 and the remote DCADO Unit 100 may learn their operations in circumstances including objects with various properties. In turn, any number of Loaders 98a can utilize the remote DCADO Unit 100 for autonomous operation in circumstances including objects with various properties. For example, multiple operators (i.e. Users 50, etc.) may operate their Loaders 98a that are configured to transmit their operations in circumstances including objects with various properties to a remote DCADO Unit 100. Such remote DCADO Unit 100 enables learning of the operators' collective knowledge of operating Loaders 98 in circumstances including objects with various properties. Any number of Loaders 98 can utilize such collective knowledge comprised in the remote DCADO Unit 100 for their autonomous operation. Any of the disclosed elements such as Artificial Intelligence Unit 110, Knowledgebase 530, and/or others may reside on Server 96, and any combination of local and remote elements can be implemented in alternate embodiments.

In some embodiments, Loader 98a may include or be coupled to a plurality of Sensors 92 and/or their corresponding Object Processing Units 93. In one example, multiple Sensors 92 may detect objects and/or their properties from different angles or on different sides of Loader 98a. In another example, one or more Sensors 92 may be placed on different sub-devices, sub-systems, or elements of Loader 98a. For instance, one Sensor 92 may be placed on the roof of Loader 98a, another Sensor 92 may be placed on the arm of Loader 98a, and an additional Sensor 92 may be placed on the bucket of Loader 98a. In some designs where multiple Sensors 92 are placed on different sub-devices, sub-systems, or elements of Loader 98a, multiple DCADO Units 100 can be utilized (i.e. one DCADO Unit 100 for each Sensor 92 or group of Sensors 92 and/or their corresponding Object Processing Units 93, etc.). In such designs, as User 50 operates Loader 98a in circumstances including objects with various properties, a particular DCADO Unit 100 may learn operations of Loader's 98a sub-device, sub-system, or element in these circumstances by correlating Collections of Object Representations 525 representing Objects 615 detected by Sensor 92 on the sub-device, sub-system, or element assigned to the DCADO Unit 100 with one or more Instruction Sets 526 used or executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element. The learning and/or decision making in Loader's 98a operation can, therefore, be performed per individual sub-device, sub-system, or element. In other designs where multiple Sensors 92 are placed on different sub-devices, sub-systems, or elements of Loader 98a, as User 50 operates Loader 98a in circumstances including objects with various properties, a single DCADO Unit 100 may learn Loader's 98a operations in these circumstances by correlating collective Collections of Object Representations 525 representing Objects 615 detected by Sensors 92 on the sub-devices, sub-systems, or elements with one or more Instruction Sets 526 used or executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element.

In some embodiments, Loader 98a may include a plurality of Logic Circuits 250, Processors 11, Application Programs 18, and/or other processing elements. In some aspects, one or more sub-devices, sub-systems, or elements of Loader 98a may be controlled by different processing elements. For example, one Processor 11 (i.e. including any Application Programs 18 running thereon, etc.) may control the moving system (i.e. drivetrain, powertrain, etc.) of Loader 98a, one Logic Circuit 250 may control an arm of Loader 98a, and an additional Logic Circuit 250 may control a bucket of Loader 98a. In some designs where multiple processing elements are utilized, multiple DCADO Units 100 can also be utilized (i.e. one DCADO Unit 100 for each processing element, etc.). In such designs, as User 50 operates Loader 98a in circumstances including objects with various properties, a particular DCADO Unit 100 may learn Loader's 98a operations in these circumstances by correlating Collections of Object

Representations 525 with one or more Instruction Sets 526 used or executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element assigned to the DCADO Unit 100. The learning and/or decision making in Loader's 98a operation can, therefore, be performed per individual processing element. In other designs where multiple processing elements are utilized, as User 50 operates Loader 98a in circumstances including objects with various properties, a single DCADO Unit 100 may learn Loader's 98a operations in these circumstances by correlating Collections of Object Representations 525 with collective Instruction Sets 526 used or executed by a plurality of Logic Circuits 250, Processors 11, Application Programs 18, and/or other processing elements.

In some embodiments, a combination of DCADO Unit 100 and other systems and/or techniques can be utilized to implement Loader's 98a operation. In one example, DCADO Unit 100 may be a primary or preferred system for implementing Loader's 98a operation. While operating autonomously under the control of DCADO Unit 100, Loader 98a may encounter a circumstance including objects with various properties that has not been encountered or learned before. In such situations, User 50 and/or non-DCADO system may take control of Loader's 98a operation. DCADO Unit 100 may take control again when Loader 98a encounters a previously learned circumstance including objects with various properties. Naturally, DCADO Unit 100 can learn Loader's 98a operation in the circumstances while User 50 and/or non-DCADO system is in control of Loader 98a, thereby reducing or eliminating the need for future involvement of User 50 and/or non-DCADO system. For instance, one User 50 can control or assist in controlling multiple Loaders 98a comprising DCADO Units 100. In such instances, User 50 can control or assist in controlling a Loader 98a that may encounter a circumstance including objects with various properties that has not been encountered or learned before while the Loaders 98a operating in previously learned circumstances can operate autonomously. In another example, User 50 and/or non-DCADO system may be a primary or preferred system for implementing Loader's 98a operation. While operating under the control of User 50 and/or non-DCADO system, User 50 and/or non-DCADO system may release control to DCADO Unit 100 for any reason (i.e. User 50 gets tired or distracted, non-DCADO system gets stuck or cannot make a decision, etc.), at which point Loader 98a can be controlled by DCADO Unit 100. In some designs, DCADO Unit 100 may take control in certain special circumstances including objects with various properties where DCADO Unit 100 may offer superior performance even though User 50 and/or non-DCADO system may generally be preferred. Once Loader 98a leaves such special circumstances, DCADO Unit 100 may release control to User 50 and/or non-DCADO system. In general, DCADO Unit 100 can take control from, share control with, or release control to User 50, non-DCADO system, and/or other system or process at any time, in any circumstances, and remain in control for any period of time as needed.

In some embodiments, DCADO Unit 100 may control one or more sub-devices, sub-systems, or elements of Loader 98a while User 50 and/or non-DCADO system may control other one or more sub-devices, sub-systems, or elements of Loader 98a. For example, User 50 and/or non-DCADO system may control the moving system (i.e. drivetrain, powertrain, etc.) of Loader 98a, while DCADO Unit 100 may control an arm and bucket of Loader 98a. Any other combination of controlling various sub-devices, sub-systems, or elements of Loader 98a by DCADO Unit 100 and User 50 and/or non-DCADO system can be implemented.

Referring to Fig. 39, in some exemplary embodiments, Device 98 may be or include Boat 98b. Boat 98b may be operated by User 50 in person or remotely. Boat 98b may include or be coupled to one or more Sensors 92

(i.e. collectively referred to as Sensor 92, etc.) such as Camera 92a, Microphone 92b, Lidar 92c, Radar 92d, Sonar 92e, etc. and/or Object Processing Unit 93 that can detect Objects 615ba-615bd, and/or other elements or information in Boat's 98b surrounding. Object Processing Unit 93 may include Picture Recognizer 94a, Sound Recognizer 94b, Lidar Processing Unit 94c, Radar Processing Unit 94d, Sonar Processing Unit 94e, and/or other elements or functionalities as applicable. Object Processing Unit 93 may create or generate one or more (i.e. stream, etc.) Collections of Object Representations 525 comprising Object Representations 625, Object Properties 630, and/or other elements or information representing Objects 615 detected in Boat's 98b surrounding. Boat 98b may also include or be controlled by Logic Circuit 250 (i.e. microcontroller, etc.), Processor 11 (i.e. including any Application Program 18 running thereon, etc.), and/or other processing element that receives User's 50 (i.e. operator's, etc.) operating directions and causes desired operations with Boat 98b such as moving, maneuvering, and/or other operations. User 50 can interact with Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element through inputting operating directions via Human-machine Interface 23 such as one or more steering wheels, levers, pedals, buttons, or other input devices. For instance, responsive to User's 50 manipulating a steering wheel and one or more levers, Logic Circuit 250 or Processor 11 may cause one or more motors or other actuators to move or maneuver Boat 98b. Boat 98b may also include or be coupled to DCADO Unit 100. DCADO Unit 100 may be embedded (i.e. integrated, etc.) into or coupled to Boat's 98b Logic Circuit 250, Processor 11, and/or other processing element. DCADO Unit 100 may also be a program embedded (i.e. integrated, etc.) into or interfaced with Application Program 18 running on Processor 11 and/or other processing element. DCADO Unit 100 can obtain Instruction Sets 526 used or executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element. In some aspects, Instruction Sets 526 may include one or more inputs into or outputs from Boat's 98b Logic Circuit 250 (i.e. microcontroller, etc.). In other aspects, Instruction Sets 526 may include one or more instruction sets from Boat's 98b Processor's 11 registers or other components. In further aspects, Instruction Sets 526 may include one or more instruction sets used or executed in Application Program 18. DCADO Unit 100 may also optionally obtain any Extra Info 527 (i.e. time, location, computed, contextual, and/or other information, etc.) related to Boat's 98b operation. As User 50 operates Boat 98b in circumstances including objects with various properties as shown, DCADO Unit 100 may learn Boat's 98b operations in these circumstances by correlating Collections of Object Representations 525 representing Objects 615 detected in Boat's 98b surrounding with one or more Instruction Sets 526 used or executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element. Any Extra Info 527 related to Boat's 98b operation may also optionally be correlated with Collections of Object Representations 525. DCADO Unit 100 may store this knowledge into Knowledgebase 530 (i.e. Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.). In the future, DCADO Unit 110 may compare incoming Collections of Object Representations 525 representing Objects 615 detected in Boat's 98b surrounding with previously learned Collections of Object Representations 525 including optionally using any Extra Info 527 for enhanced decision making. If substantially similar or at least a partial match is found or determined, the Instruction Sets 526 correlated with the previously learned Collections of Object Representations 525 can be autonomously executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element, thereby enabling autonomous operation of Boat 98b in similar circumstances as in previously learned ones. For instance, Boat 98b comprising DCADO Unit 100 may learn User 50-directed moving, maneuvering, and/or other operations in

a circumstance that includes Fishing Boat 615ba, Lighthouse 615bb, Sailboat 615bc, Cruise Ship 615bd, and/or other Objects 615 among which Boat 98b may need to maneuver. In the future, when a circumstance that includes Objects 615 with similar Object Properties 630 is encountered, Boat 98b may implement moving, maneuvering, and/or other operations autonomously. In some aspects, the shore (not enumerated) or any part thereof (i.e. cliff, ridge, beach, etc.) may be detected as an Object 615 itself, which may then be learned and used in autonomous operation of Boat 98b.

Referring to Fig. 40, in some exemplary embodiments, an Area of Interest 450 can be utilized. In one example, Area of Interest 450 may include a radial, circular, elliptical, or other such area around Boat 98b. In another example, Area of Interest 450 may include a triangular, rectangular, octagonal, or other such area around Boat 98b. In a further example, Area of Interest 450 may include a spherical, cubical, pyramid-like, or other such area around Boat 98b as applicable to 3D space. Any other Area of Interest 450 shape can be utilized depending on implementation. The shape and/or size of Area of Interest 450 can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. Utilizing Area of Interest 450 enables DCADO Unit 100 to focus on Boat's 98b immediate surrounding, thereby avoiding extraneous detail in the rest of the surrounding. In some aspects, Area of Interest 450 can be subdivided into sub-areas (i.e. sub-circles, sub-rectangles, sub-spheres, etc.). Sub-areas can be used to classify the surrounding by distance from Boat 98b. For example, the surrounding closer to Boat 98b may be more important and may be assigned higher importance index or weight. As User 50 operates Boat 98b in circumstances including objects with various properties as shown, DCADO Unit 100 may learn Boat's 98b operations in these circumstances by correlating Collections of Object Representations 525 representing Objects 615 detected in Area of Interest 450 around Boat 98b with one or more Instruction Sets 526 used or executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element. Any Extra Info 527 related to Boat's 98b operation may also optionally be correlated with Collections of Object Representations 525. DCADO Unit 100 may store this knowledge into Knowledgebase 530 (i.e. Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.). In the future, DCADO Unit 110 may compare incoming Collections of Object Representations 525 representing Objects 615 detected in Area of Interest 450 around Boat 98b with previously learned Collections of Object Representations 525 including optionally using any Extra Info 527 for enhanced decision making. If substantially similar or at least a partial match is found or determined, the Instruction Sets 526 correlated with the previously learned Collections of Object Representations 525 can be autonomously executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element, thereby enabling autonomous operation of Boat 98b in similar Areas of Interest 450 as in previously learned ones. For instance, Boat 98b comprising DCADO Unit 100 may learn User 50-directed moving, maneuvering, and/or other operations in an Area of Interest 450 that includes Fishing Boat 615ba, Lighthouse 615bb, Cruise Ship 615bd, and/or other Objects 615 among which Boat 98b may need to maneuver. In the future, when an Area of Interest 450 that includes Objects 615 with similar Object Properties 630 is encountered, Boat 98b may implement moving, maneuvering, and/or other operations autonomously.

The features, functionalities, and embodiments described with respect to Loader 98a and Boat 98b can be implemented in any situation where Device 98 may need to autonomously maneuver among, interact with, or perform other operations relative to objects in its surrounding. Therefore, the features, functionalities, and

embodiments described with respect to Loader 98a and Boat 98b can similarly be implemented on any computing enabled machine such as a bulldozer, an excavator, a crane, a forklift, a truck, a construction machine, an assembly machine, an object handling machine, an object dispensing machine, a sorting machine, a restocking machine, an industrial machine, an agricultural machine, a harvesting machine, a building control system, a home or other appliance, a toy, a robot, a tank, an aircraft, a vessel, a submarine, a ground vehicle, an aerial vehicle, an aquatic vehicle, and/or other computing-enabled machine or system.

In yet some exemplary embodiments, Device 98 may be or include a control device such as a thermostat, control panel, remote or other controller, and/or other control device. For instance, a thermostat comprising DCADO Unit 100 may learn User's 50 setting temperature of an air conditioning system controlled by the thermostat in a circumstance that includes User 50 and/or other persons entering or being present in a room. In the future, when a circumstance that includes User 50 and/or other persons entering or being present in the room is encountered, thermostat may implement setting temperature of the air conditioning system autonomously. In some aspects, a control device may be included in the device being controlled (i.e. control panel of an oven, refrigerator, fixture, etc.). In other aspects, a control device may be separate from the device being controlled (i.e. remote controller of a television device, etc.). In yet further exemplary embodiments, Device 98 may be or include a mobile computer such as a smartphone, tablet, and/or other mobile computer. For instance, a smartphone comprising DCADO Unit 100 may learn User 50-directed playing a music file, setting a vibrate mode, and/or other operations in a circumstance that includes objects with various properties. In the future, when a circumstance that includes objects with similar properties is encountered, smartphone may implement playing music file, setting vibrate mode, and/or other operations autonomously. In general, Device 98 may be or include any movable, stationary, or other device. One of ordinary skill in art will understand that Device 98 may be or include any device that can implement and/or benefit from the functionalities described herein.

It must be noted that as used herein and in the appended claims, the singular forms "a", "an", and "the" include plural referents unless the context clearly dictates otherwise.

A number of embodiments have been described herein. While this disclosure contains many specific implementation details, these should not be construed as limitations on the scope of the disclosure or of what may be claimed, but rather as descriptions of features specific to particular embodiments. It should be understood that various modifications can be made without departing from the spirit and scope of the disclosure. The logic flows depicted in the figures do not require the particular order shown, or sequential order, to achieve desirable results. In addition, other or additional steps, elements, or connections can be included, or some of the steps, elements, or connections can be eliminated, or a combination thereof can be utilized in the described flows, illustrations, or descriptions. Further, the various aspects of the disclosed devices, apparatuses, systems, and/or methods can be combined in whole or in part with each other to produce additional implementations. Moreover, separation of various components in the embodiments described herein should not be understood as requiring such separation in all embodiments, and it should be understood that the described components can generally be integrated together in a single product or packaged into multiple products. Accordingly, other embodiments are within the scope of the following claims.

CLAIMS

Claim 1. A system for learning and using a device's circumstances for autonomous device operating, the system implemented at least in part on one or more computing devices, the system comprising:

a processor circuit configured to execute instruction sets for operating a device;

a memory unit configured to store data;

a sensor configured to detect objects; and

an artificial intelligence unit configured to:

receive a first stream of collections of object representations, the first stream of collections of object representations including one or more collections of representations of objects detected by the sensor;

receive a first one or more instruction sets for operating the device;

learn the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device;

receive a new stream of collections of object representations, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor;

anticipate the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations; and

cause the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations performed in response to the executing by the processor circuit.

Claim 2. The system of Claim 1, wherein the receiving the first one or more instruction sets for operating the device includes a tracing of the processor circuit or a tracing of a component of the processor circuit.

Claim 3. The system of Claim 1, further comprising:

an application including the instruction sets for operating the device, the application running on the processor circuit, wherein the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from the application.

Claim 4. The system of Claim 3, wherein the receiving the first one or more instruction sets for operating the device from the application includes at least one of: a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of the application.

Claim 5. The system of Claim 3, wherein the receiving the first one or more instruction sets for operating the device from the application includes a tracing of the application.

Claim 6. The system of Claim 1, wherein the first one or more instruction sets for operating the device include one or more instruction sets that temporally correspond to the first stream of collections of object representations.

Claim 7. The system of Claim 1, wherein the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device include a knowledge of how the device operated in a circumstance.

Claim 8. The system of Claim 1, wherein the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device includes storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device into the memory unit, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device being part of a plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device stored in the memory unit.

Claim 9. The system of Claim 1, wherein at least one of: the processor circuit, the memory unit, the sensor, or the artificial intelligence unit are part of the device.

Claim 10. The system of Claim 1, wherein the artificial intelligence unit is part of, operating on, or coupled to the processor circuit.

Claim 11. A non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising:

receiving a first stream of collections of object representations, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor;

receiving a first one or more instruction sets for operating a device;

learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device;

receiving a new stream of collections of object representations, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor;

anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on

at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations; and

causing an execution of the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the execution.

Claim 12. The non-transitory computer storage medium of Claim 11, wherein the execution of the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations is performed by the one or more processor circuits or by another one or more processor circuits.

Claim 13. The non-transitory computer storage medium of Claim 11, wherein the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from an application, the application comprising instruction sets for operating the device.

Claim 14. The non-transitory computer storage medium of Claim 13, wherein the receiving the first one or more instruction sets for operating the device from the application includes a tracing of the application.

Claim 15. The non-transitory computer storage medium of Claim 11, wherein the first one or more instruction sets for operating the device include one or more instruction sets that temporally correspond to the first stream of collections of object representations.

Claim 16. A method comprising:

(a) receiving a first stream of collections of object representations by a processor circuit, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor;

(b) receiving a first one or more instruction sets for operating a device by the processor circuit;

(c) learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the processor circuit;

(d) receiving a new stream of collections of object representations by the processor circuit, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor;

(e) anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, the anticipating of (e) performed by the processor circuit;

(f) executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the executing of (f) performed in response to the anticipating of (e); and

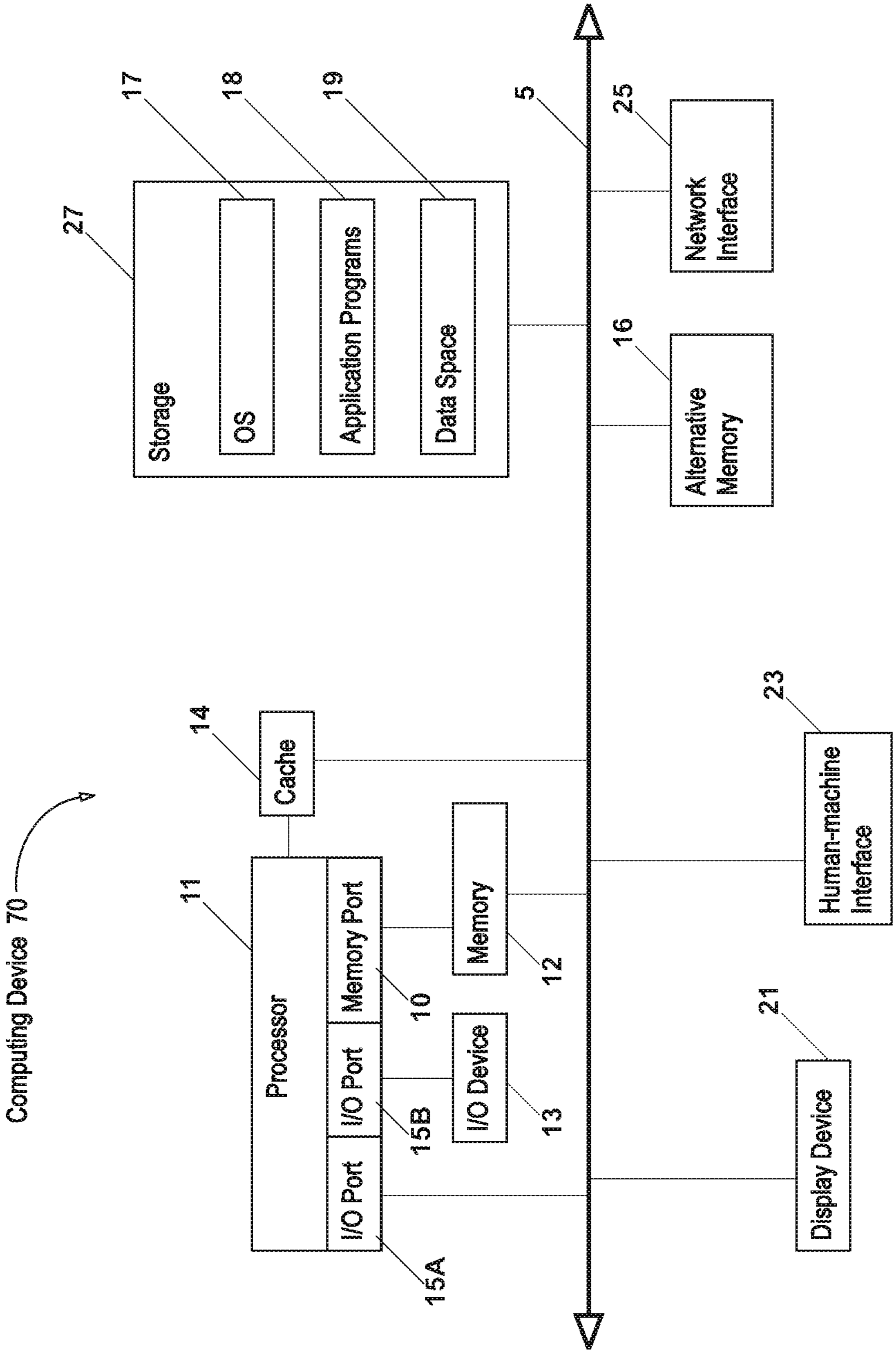
(g) performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the executing of (f).

Claim 17. The method of Claim 16, wherein the executing of (f) is performed by the processor circuit or by another processor circuit.

Claim 18. The method of Claim 16, wherein the receiving of (b) includes receiving the first one or more instruction sets for operating the device from an application, the application comprising instruction sets for operating the device.

Claim 19. The method of Claim 18, wherein the receiving the first one or more instruction sets for operating the device from the application includes a tracing of the application.

Claim 20. The method of Claim 16, wherein the first one or more instruction sets for operating the device include one or more instruction sets that temporally correspond to the first stream of collections of object representations.



Computing Device 70
Processor
11
I/O Port
15A
I/O Port
15B
Memory Port
10
Cache
14
I/O Device
13
Memory
12
5
Storage
27
OS
17
Application Programs
18
Data Space
19
Alternative Memory
16
Network Interface
25
Display Device
21
Human-machine Interface
23

FIG. 1

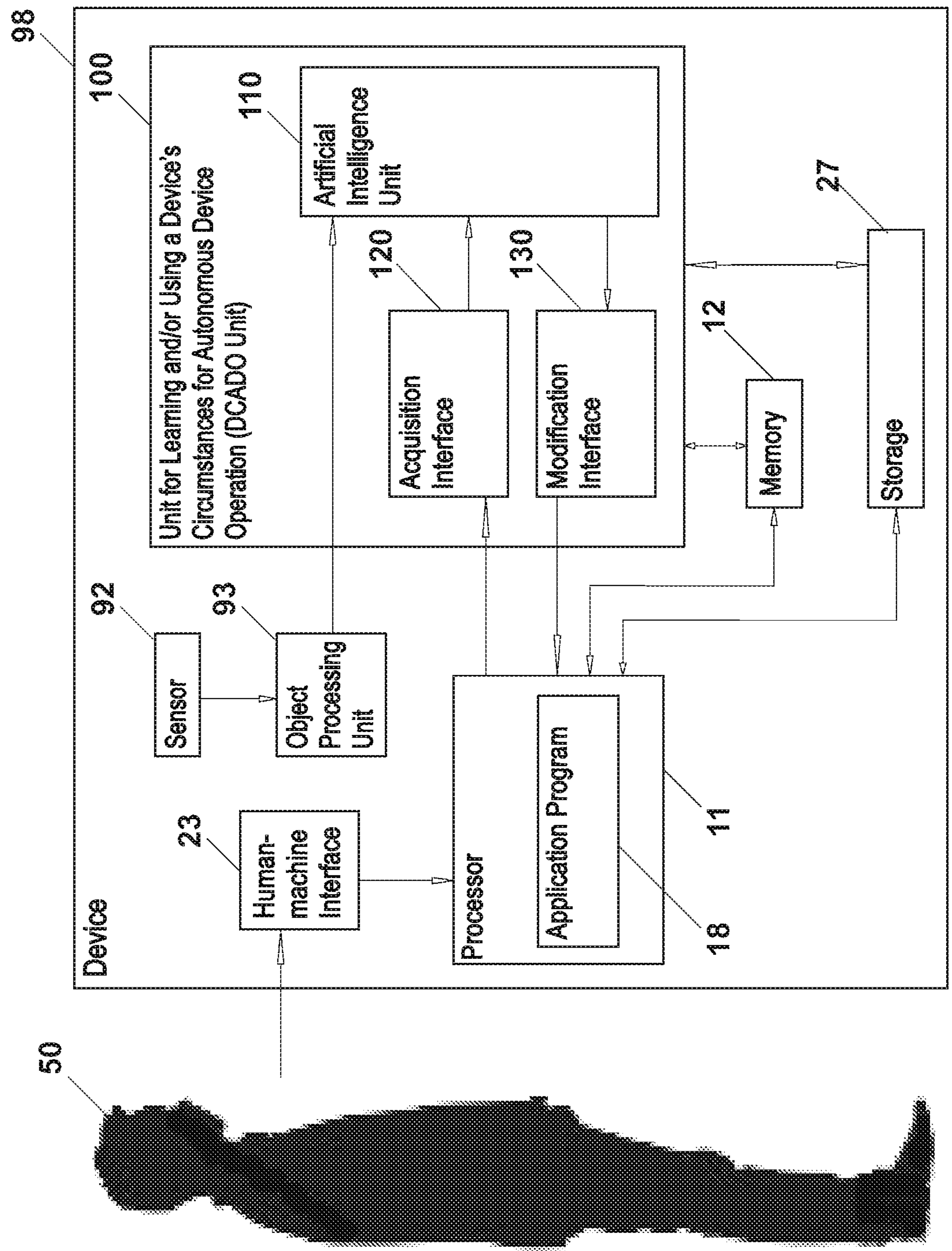

98
Device
100
Unit for Learning and/or Using a Device's Circumstances for Autonomous Device Operation (DCADO Unit)
110
Artificial Intelligence Unit
120
Acquisition Interface
130
Modification Interface
12
Memory
27
Storage
92
Sensor
93
Object Processing Unit
23
Human-machine Interface
Processor
Application Program
11
18
50

FIG. 2

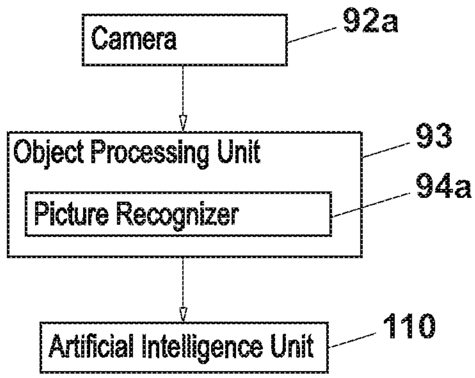


FIG. 3A

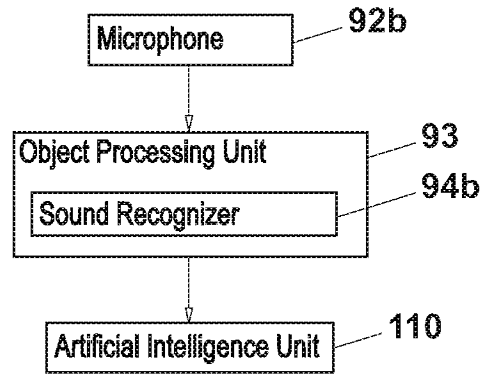


FIG. 3B

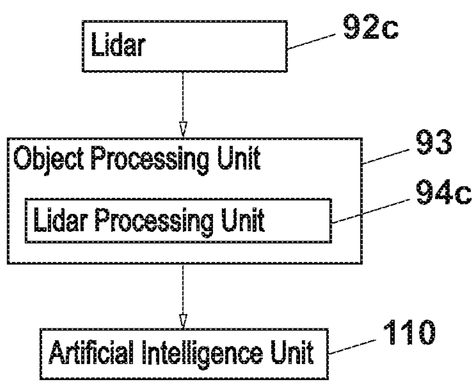


FIG. 3C

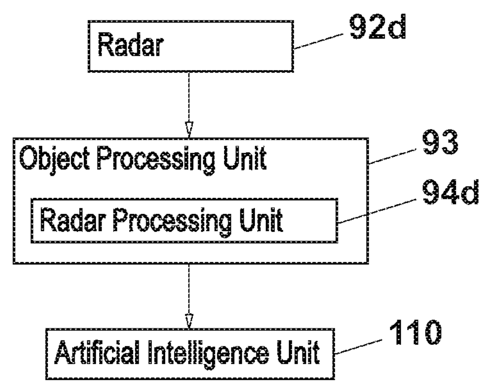


FIG. 3D

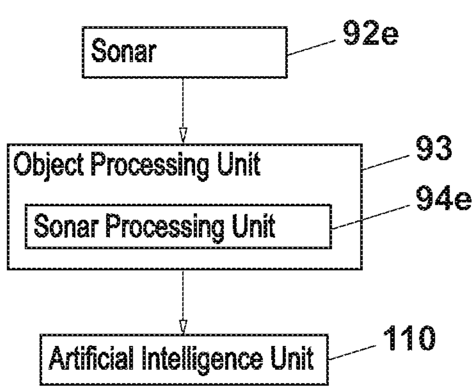
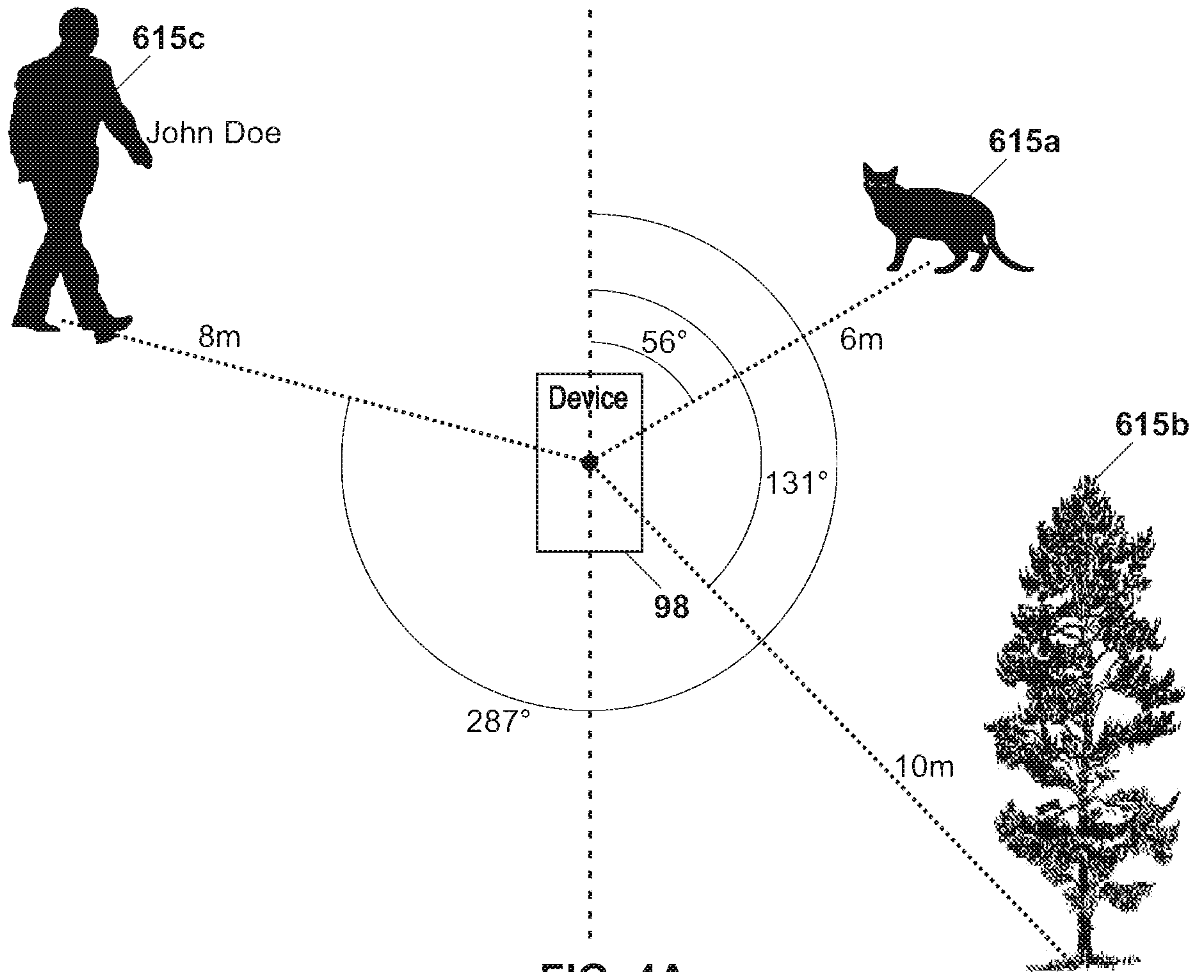
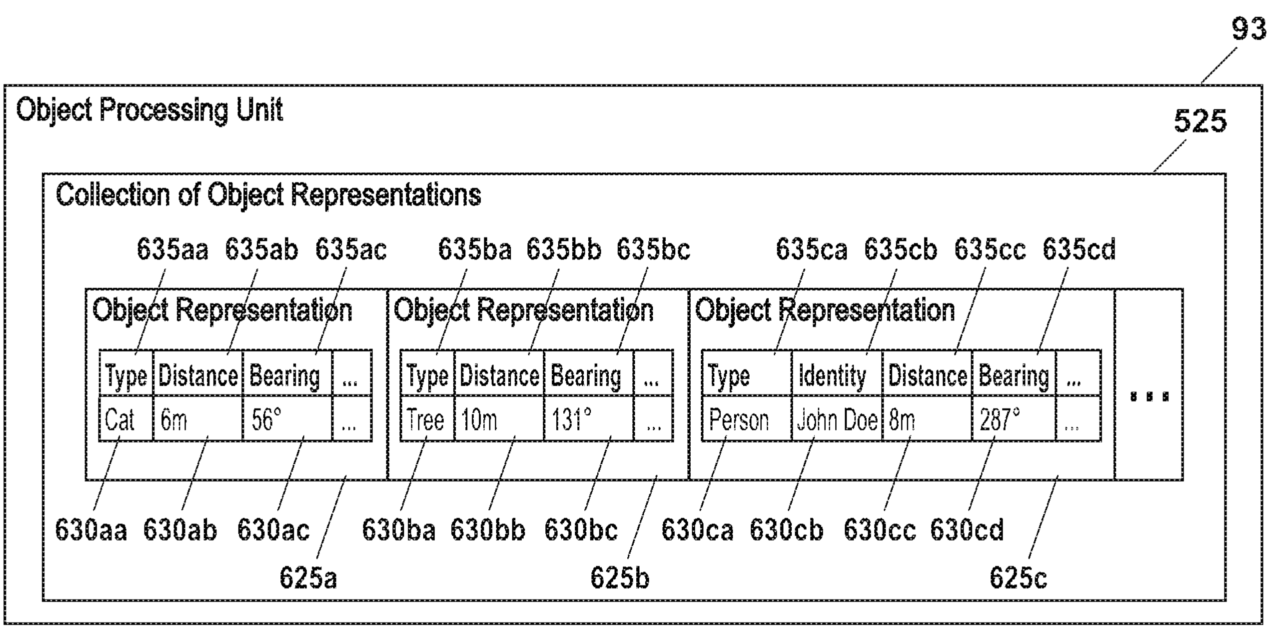


FIG. 3E



615c
John Doe
615a
8m
56°
6m
Device
615b
131°
98
287°
10m

FIG. 4A



93
Object Processing Unit
525
Collection of Object Representations
635aa 635ab 635ac
635ba 635bb 635bc
635ca 635cb 635cc 635cd
Object Representation
Type | Distance | Bearing | ...
Cat | 6m | 56° | ...
Object Representation
Type | Distance | Bearing | ...
Tree | 10m | 131° | ...
Object Representation
Type | Identity | Distance | Bearing | ...
Person | John Doe | 8m | 287° | ...
...
630aa 630ab 630ac
630ba 630bb 630bc
630ca 630cb 630cc 630cd
625a
625b
625c

FIG. 4B

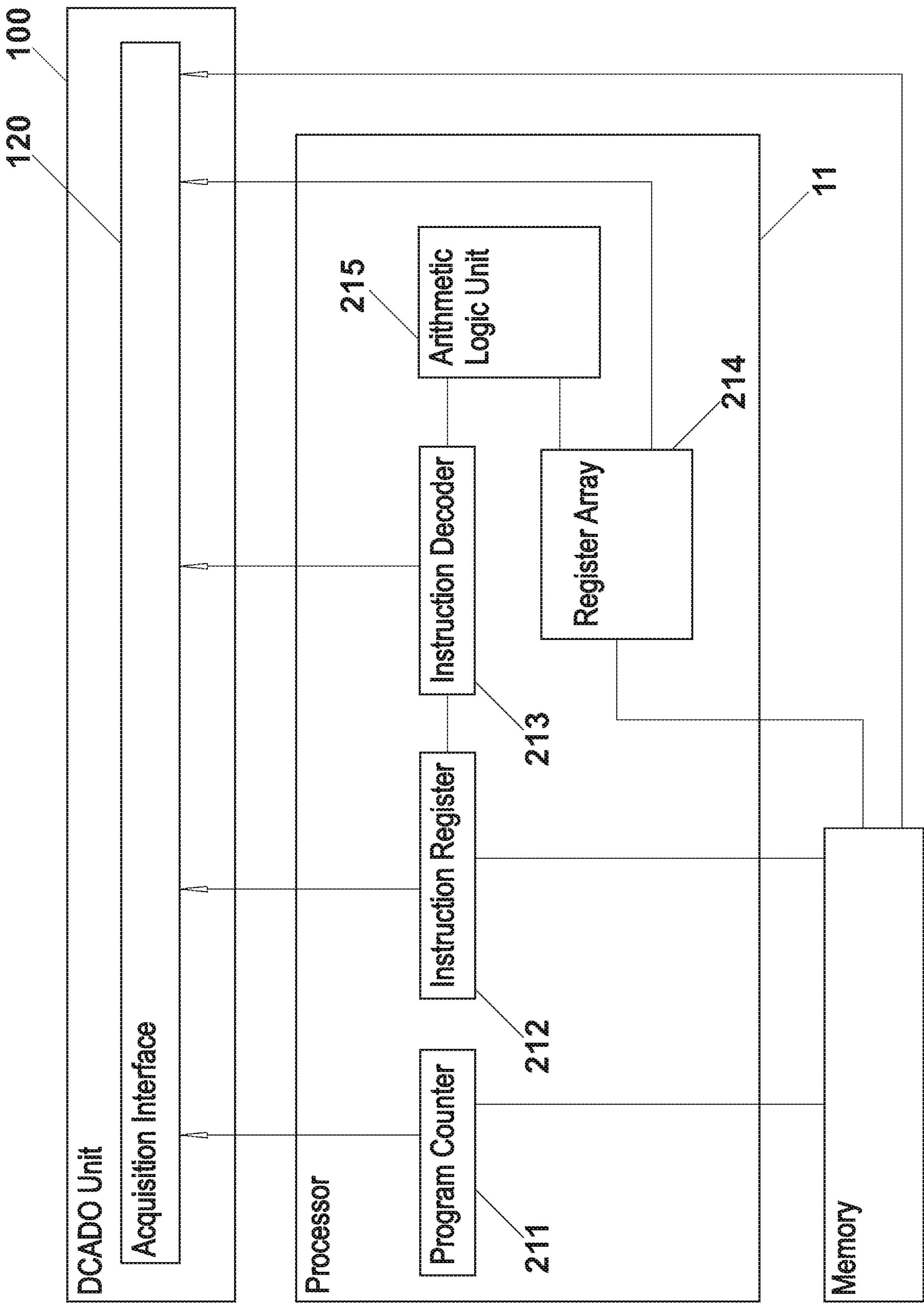

100
120
DCADO Unit
Acquisition Interface
Processor
11
215
Arithmetic Logic Unit
214
Instruction Decoder
Register Array
213
Instruction Register
212
Program Counter
211
Memory
12

FIG. 5

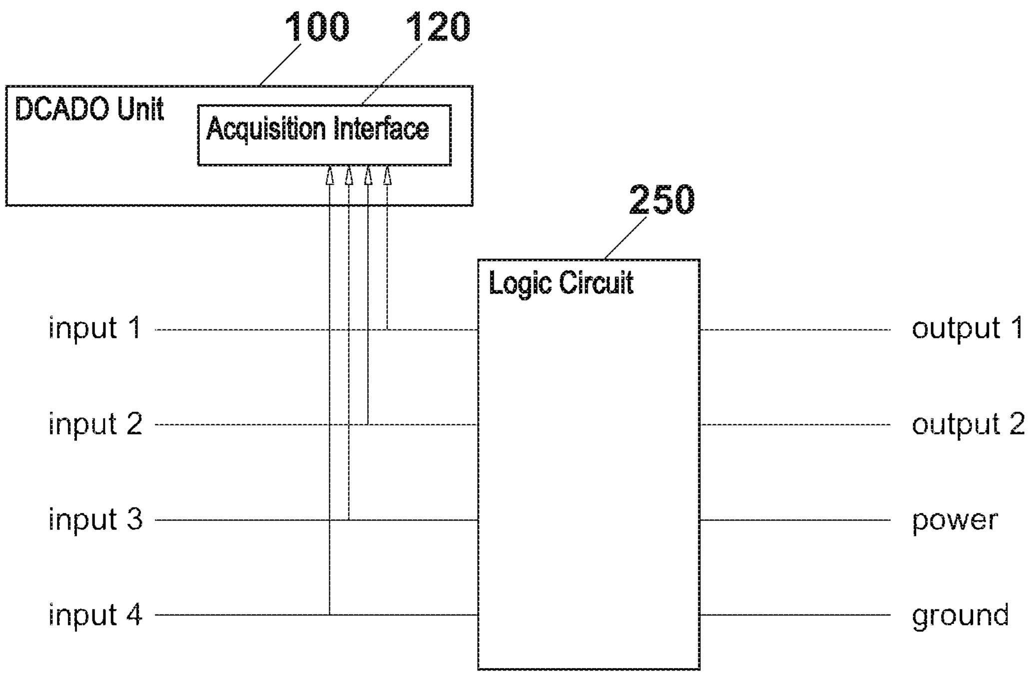

100
120
DCADO Unit
Acquisition Interface
250
Logic Circuit
input 1
input 2
input 3
input 4
output 1
output 2
power
ground

FIG. 6A

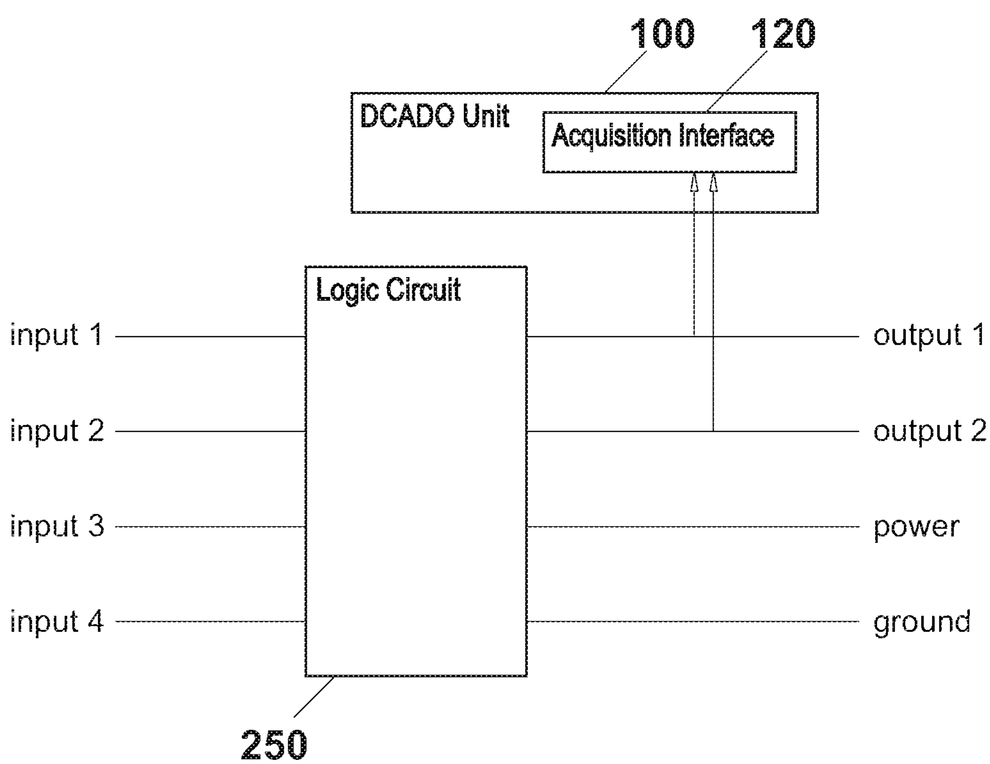

100
120
DCADO Unit
Acquisition Interface
Logic Circuit
input 1
input 2
input 3
input 4
output 1
output 2
power
ground
250

FIG. 6B

526

Instruction Set

Function (Parameter1, Parameter2, Parameter3, ...)

FIG. 7A

526

Instruction Set

SELECT Col1, Col2 ... FROM Tbl1, Tbl2 ... ;

FIG. 7B

526

Instruction Set

iconst_Value

FIG. 7C

526

Instruction Set

add Register1 Register2

FIG. 7D

526

Instruction Set

000000 00001 00010 00110 00000 100000

FIG. 7E

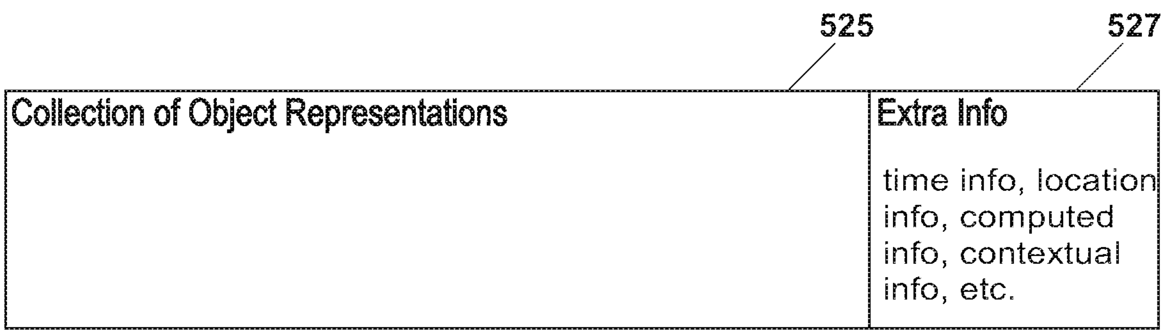

525
527
Collection of Object Representations
Extra Info
time info, location info, computed info, contextual info, etc.

FIG. 8A

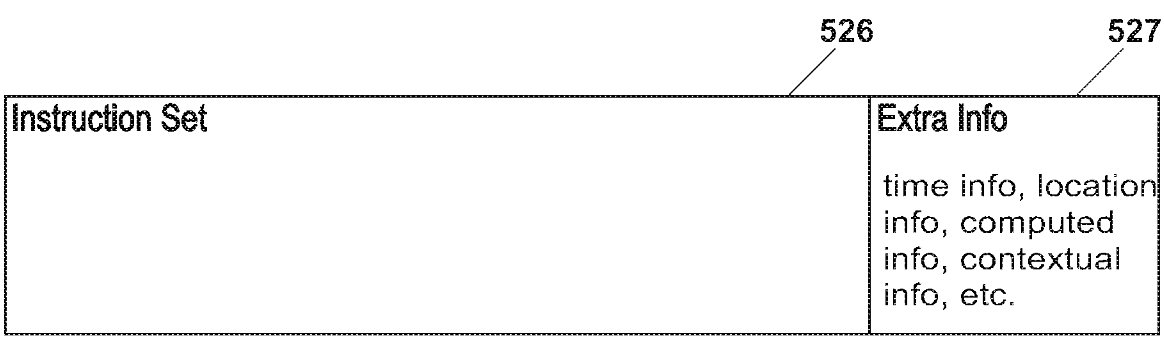

526
527
Instruction Set
Extra Info
time info, location info, computed info, contextual info, etc.

FIG. 8B

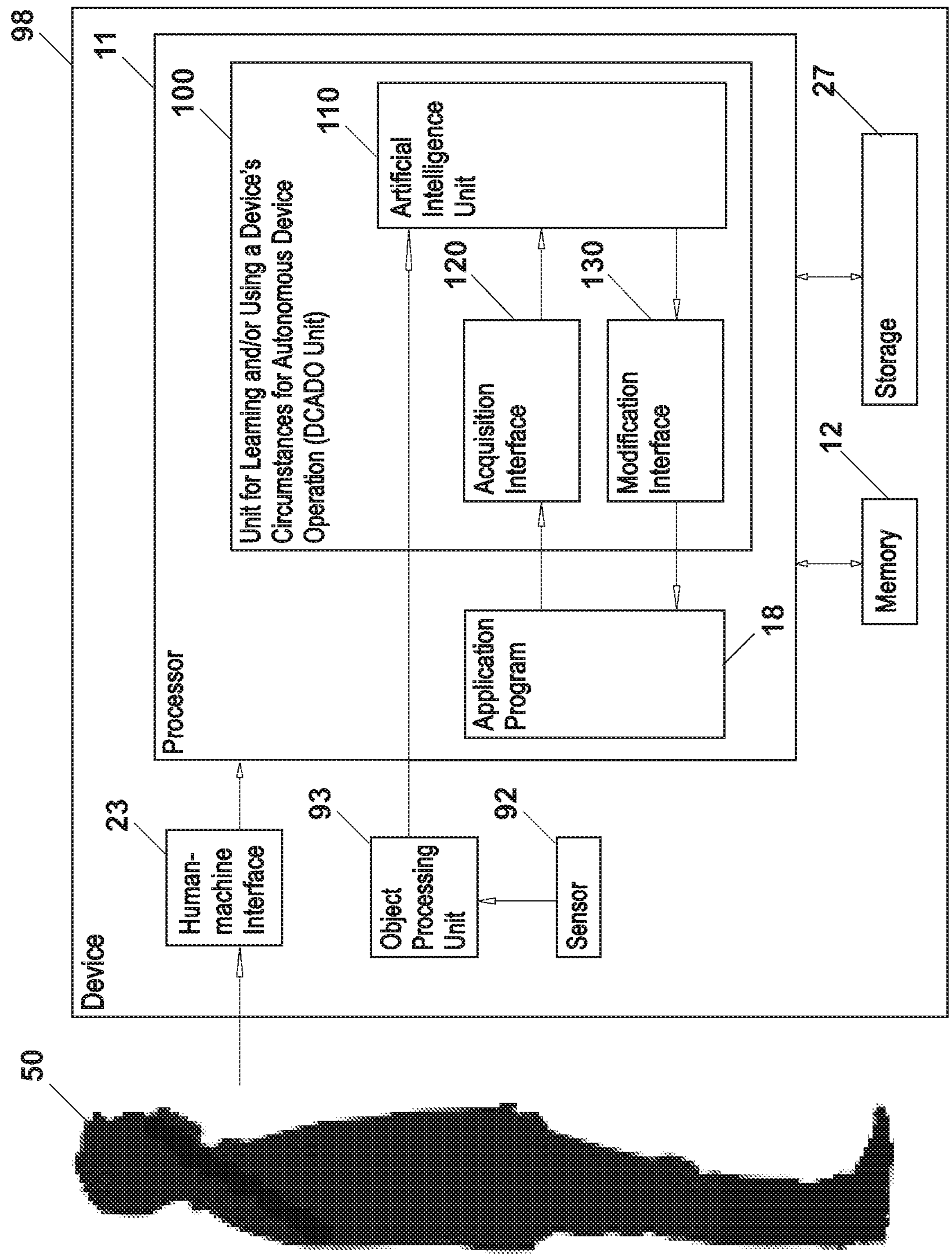

98
Device
11
Processor
100
Unit for Learning and/or Using a Device's Circumstances for Autonomous Device Operation (DCADO Unit)
110
Artificial Intelligence Unit
120
Acquisition Interface
130
Modification Interface
18
Application Program
27
Storage
12
Memory
23
Human-machine Interface
93
Object Processing Unit
92
Sensor
50

FIG. 9

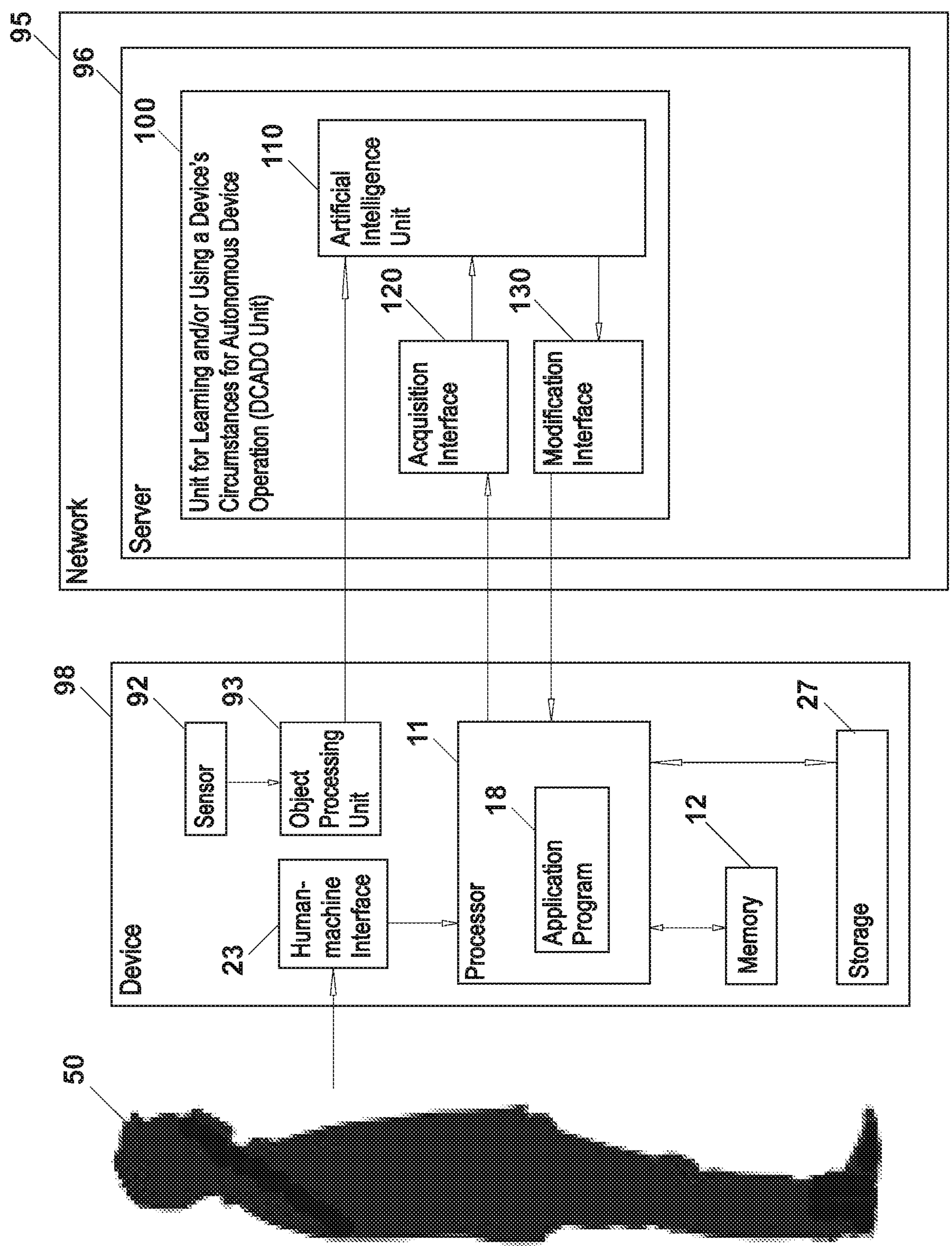
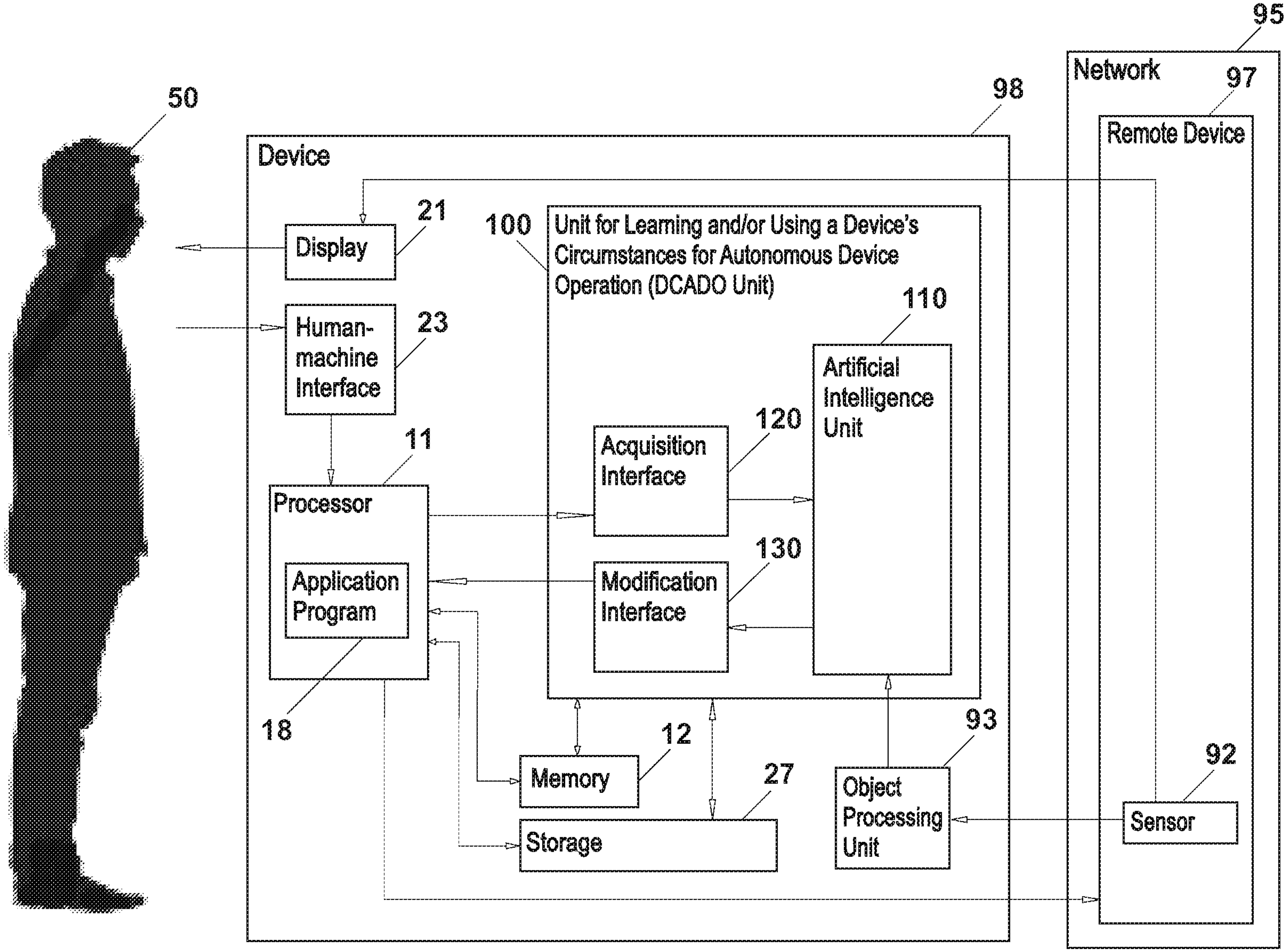

95
Network
96
Server
100
Unit for Learning and/or Using a Device's Circumstances for Autonomous Device Operation (DCADO Unit)
110
Artificial Intelligence Unit
120
Acquisition Interface
130
Modification Interface
98
Device
92
Sensor
93
Object Processing Unit
23
Human-machine Interface
11
Processor
18
Application Program
12
Memory
27
Storage
50

FIG. 10



50
Device
98
Network
95
97
Remote Device
Display
21
100
Unit for Learning and/or Using a Device's Circumstances for Autonomous Device Operation (DCADO Unit)
Human-machine Interface
23
110
Artificial Intelligence Unit
120
Acquisition Interface
11
Processor
130
Modification Interface
Application Program
18
12
Memory
27
Storage
93
Object Processing Unit
92
Sensor

FIG. 11

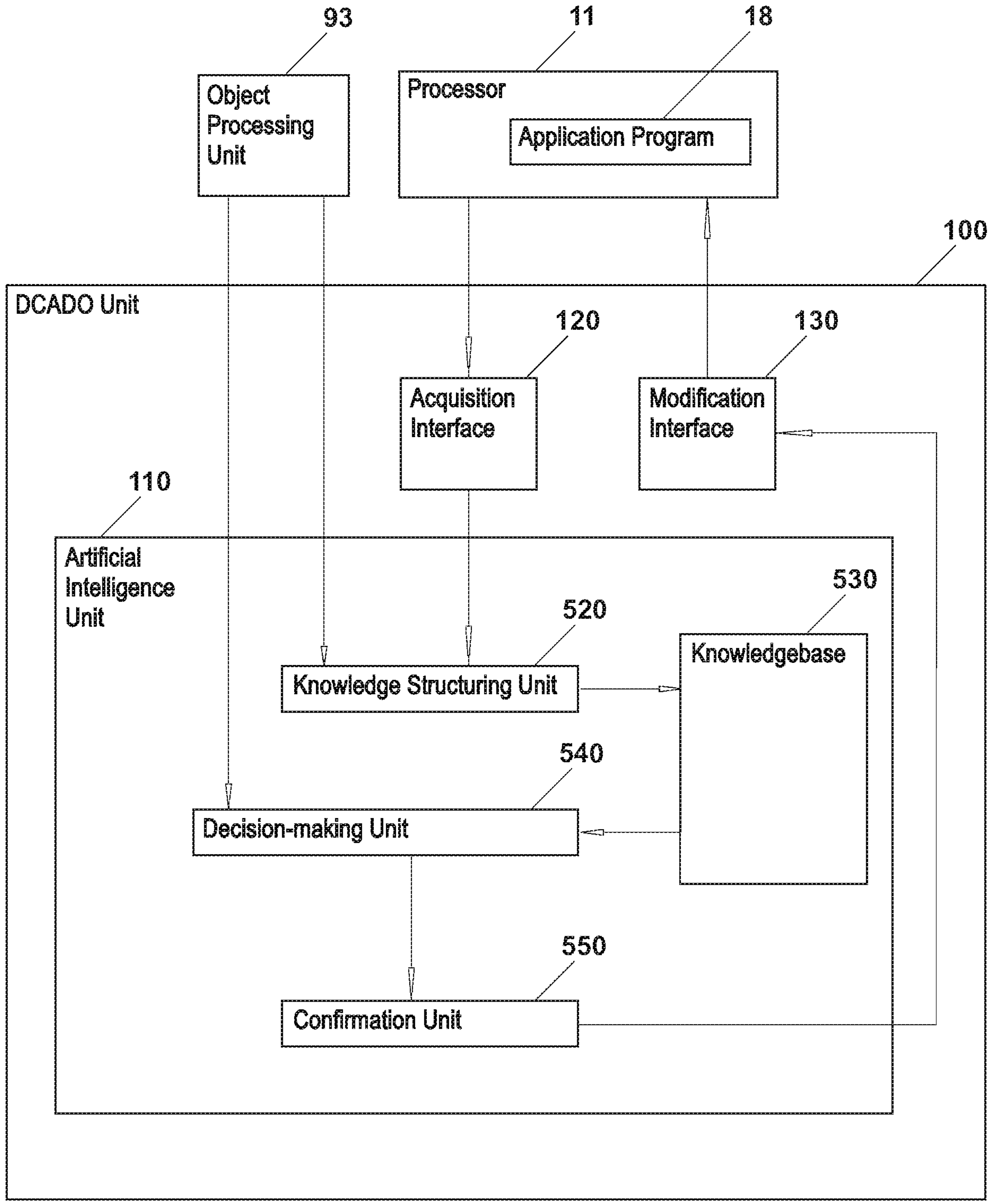
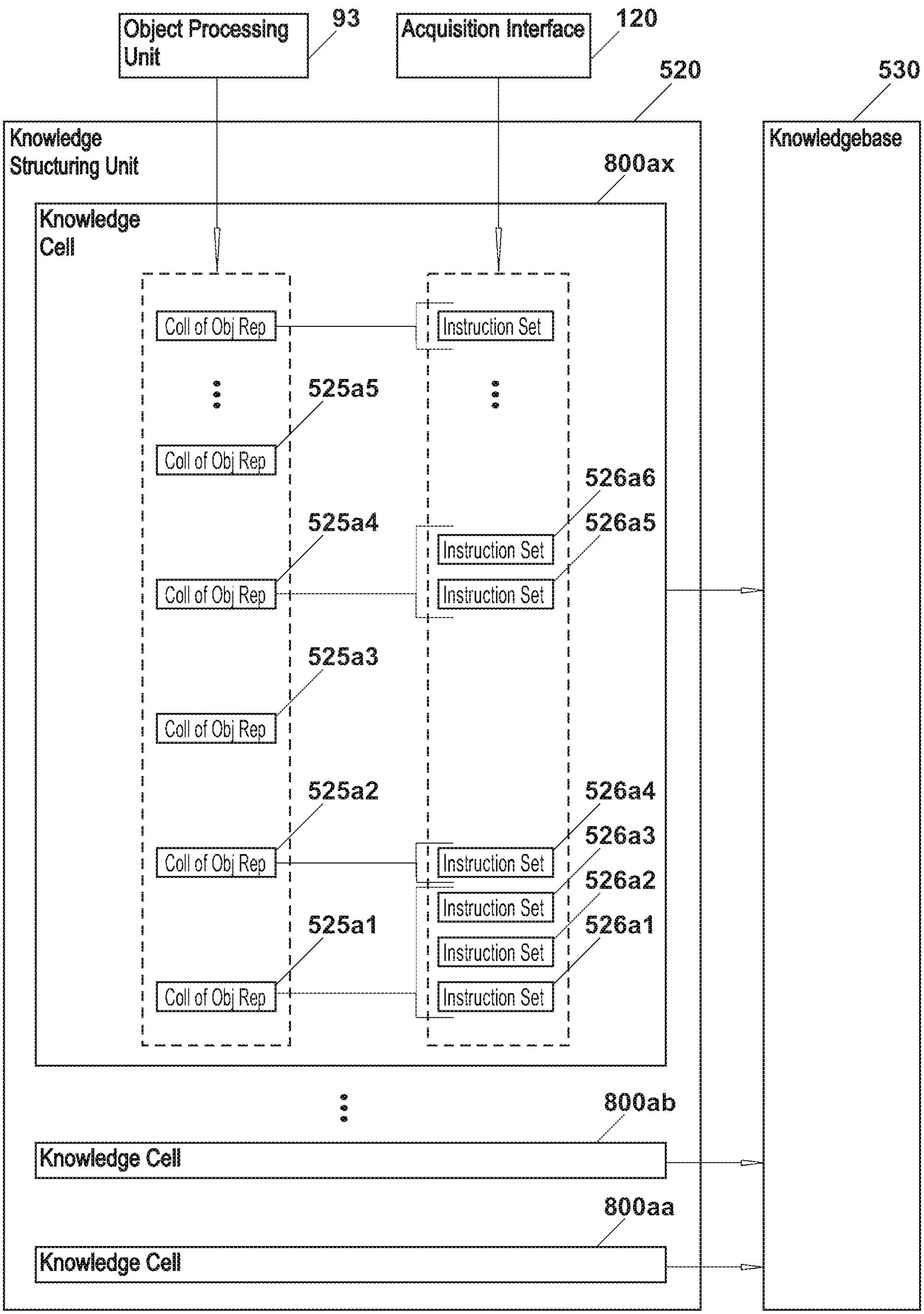
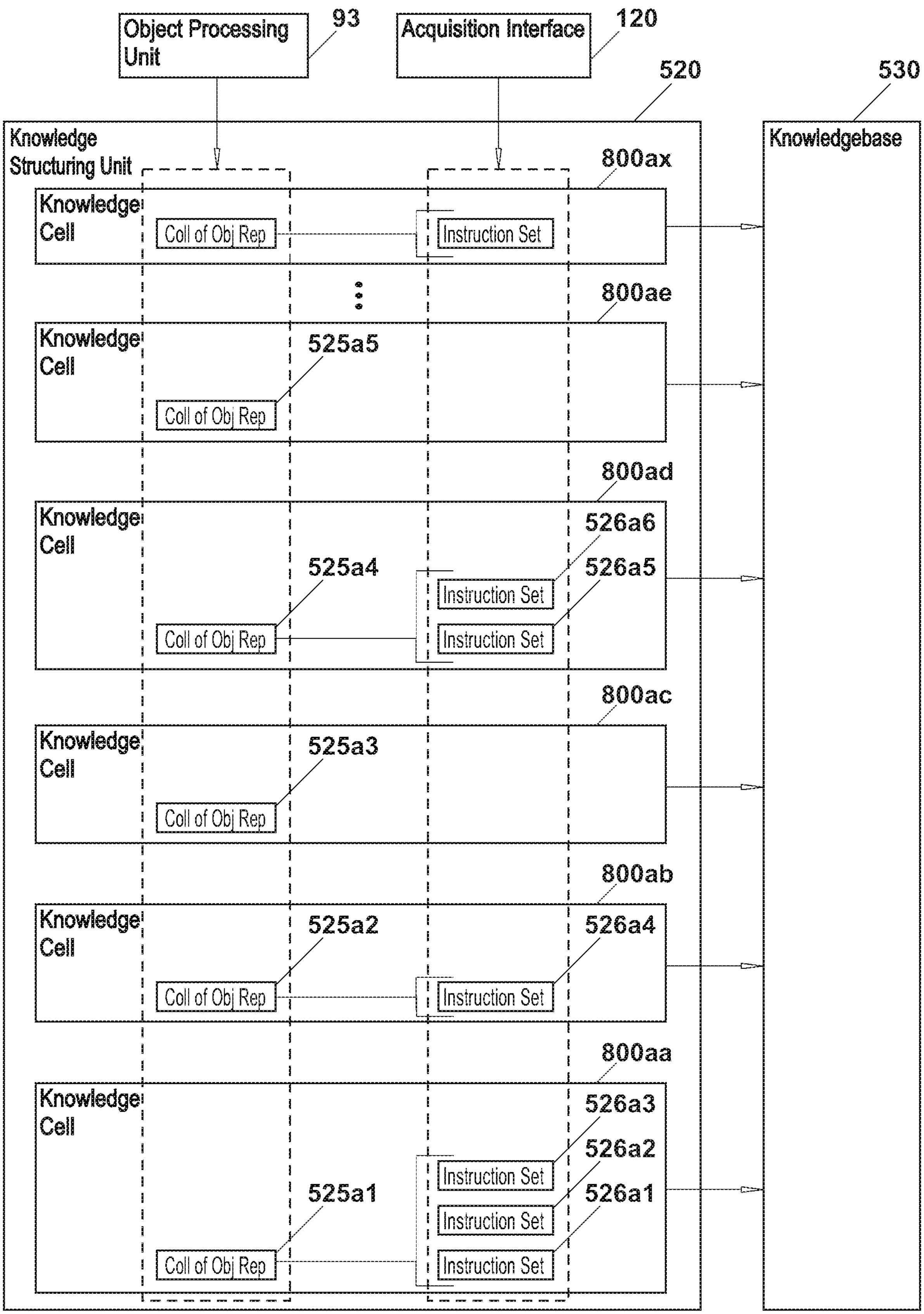

93
Object Processing Unit
11
Processor
18
Application Program
100
DCADO Unit
120
Acquisition Interface
130
Modification Interface
110
Artificial Intelligence Unit
530
Knowledgebase
520
Knowledge Structuring Unit
540
Decision-making Unit
550
Confirmation Unit

FIG. 12



Object Processing Unit
93
Acquisition Interface
120
520
530
Knowledge Structuring Unit
800ax
Knowledgebase
Knowledge Cell
Coll of Obj Rep
Instruction Set
525a5
Coll of Obj Rep
526a6
525a4
526a5
Instruction Set
Coll of Obj Rep
Instruction Set
525a3
Coll of Obj Rep
525a2
526a4
526a3
Coll of Obj Rep
Instruction Set
526a2
Instruction Set
525a1
526a1
Instruction Set
Coll of Obj Rep
Instruction Set
800ab
Knowledge Cell
800aa
Knowledge Cell

FIG. 13



Object Processing Unit
93
Acquisition Interface
120
520
530
Knowledge Structuring Unit
Knowledgebase
800ax
Knowledge Cell
Coll of Obj Rep
Instruction Set
800ae
Knowledge Cell
525a5
Coll of Obj Rep
800ad
Knowledge Cell
526a6
525a4
526a5
Instruction Set
Coll of Obj Rep
Instruction Set
800ac
Knowledge Cell
525a3
Coll of Obj Rep
800ab
Knowledge Cell
525a2
526a4
Coll of Obj Rep
Instruction Set
800aa
Knowledge Cell
526a3
526a2
Instruction Set
525a1
526a1
Instruction Set
Coll of Obj Rep
Instruction Set

FIG. 14

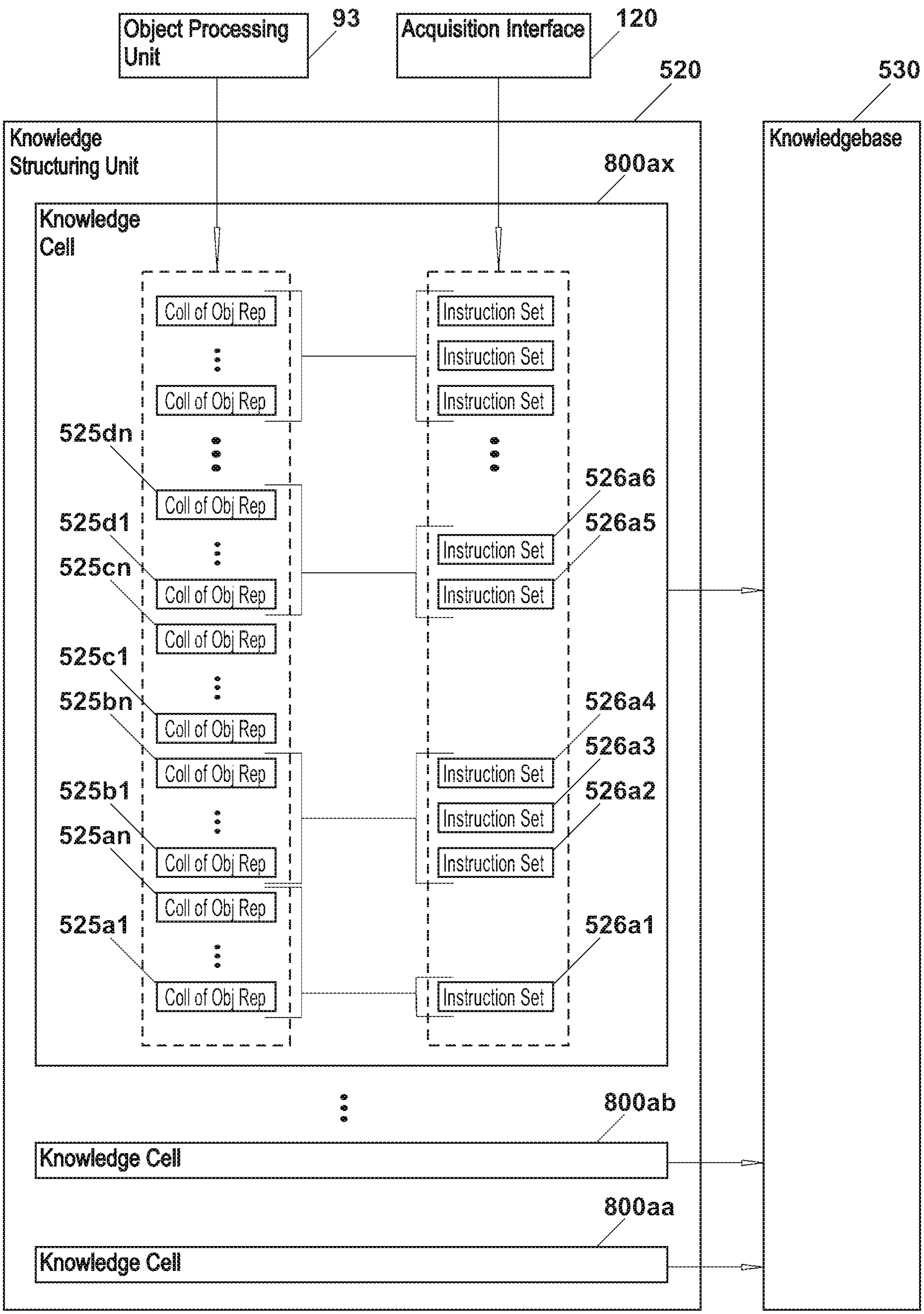
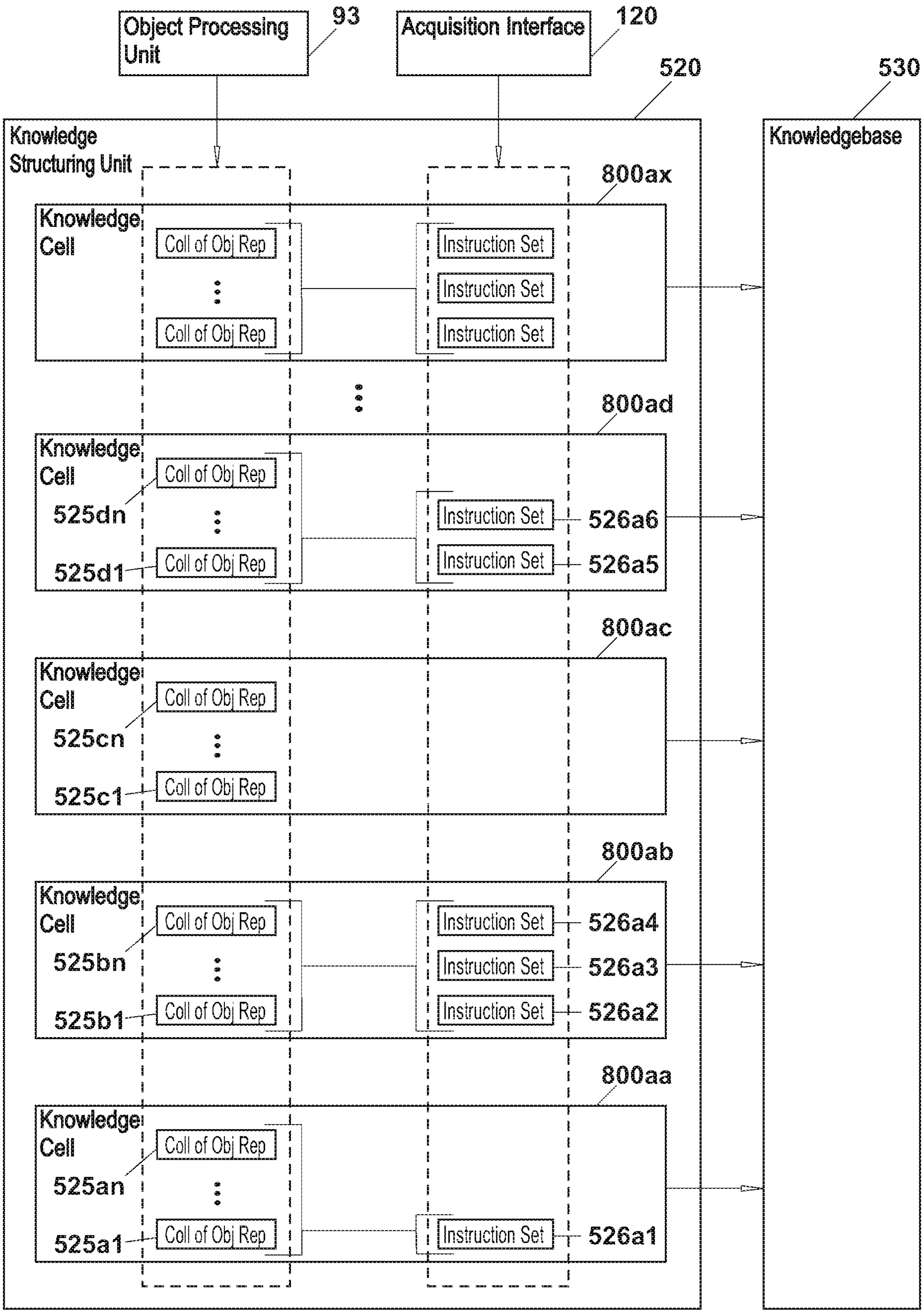

Object Processing Unit
93
Acquisition Interface
120
520
530
Knowledge Structuring Unit
Knowledgebase
800ax
Knowledge Cell
Coll of Obj Rep
Instruction Set
525dn
525d1
525cn
525c1
525bn
525b1
525an
525a1
526a6
526a5
526a4
526a3
526a2
526a1
800ab
Knowledge Cell
800aa
Knowledge Cell

FIG. 15



Object Processing Unit
93
Acquisition Interface
120
520
530
Knowledge Structuring Unit
Knowledgebase
800ax
Knowledge Cell
Coll of Obj Rep
Coll of Obj Rep
Instruction Set
Instruction Set
Instruction Set
800ad
Knowledge Cell
525dn
525d1
Coll of Obj Rep
Coll of Obj Rep
Instruction Set
Instruction Set
526a6
526a5
800ac
Knowledge Cell
525cn
525c1
Coll of Obj Rep
Coll of Obj Rep
800ab
Knowledge Cell
525bn
525b1
Coll of Obj Rep
Coll of Obj Rep
Instruction Set
Instruction Set
Instruction Set
526a4
526a3
526a2
800aa
Knowledge Cell
525an
525a1
Coll of Obj Rep
Coll of Obj Rep
Instruction Set
526a1

FIG. 16

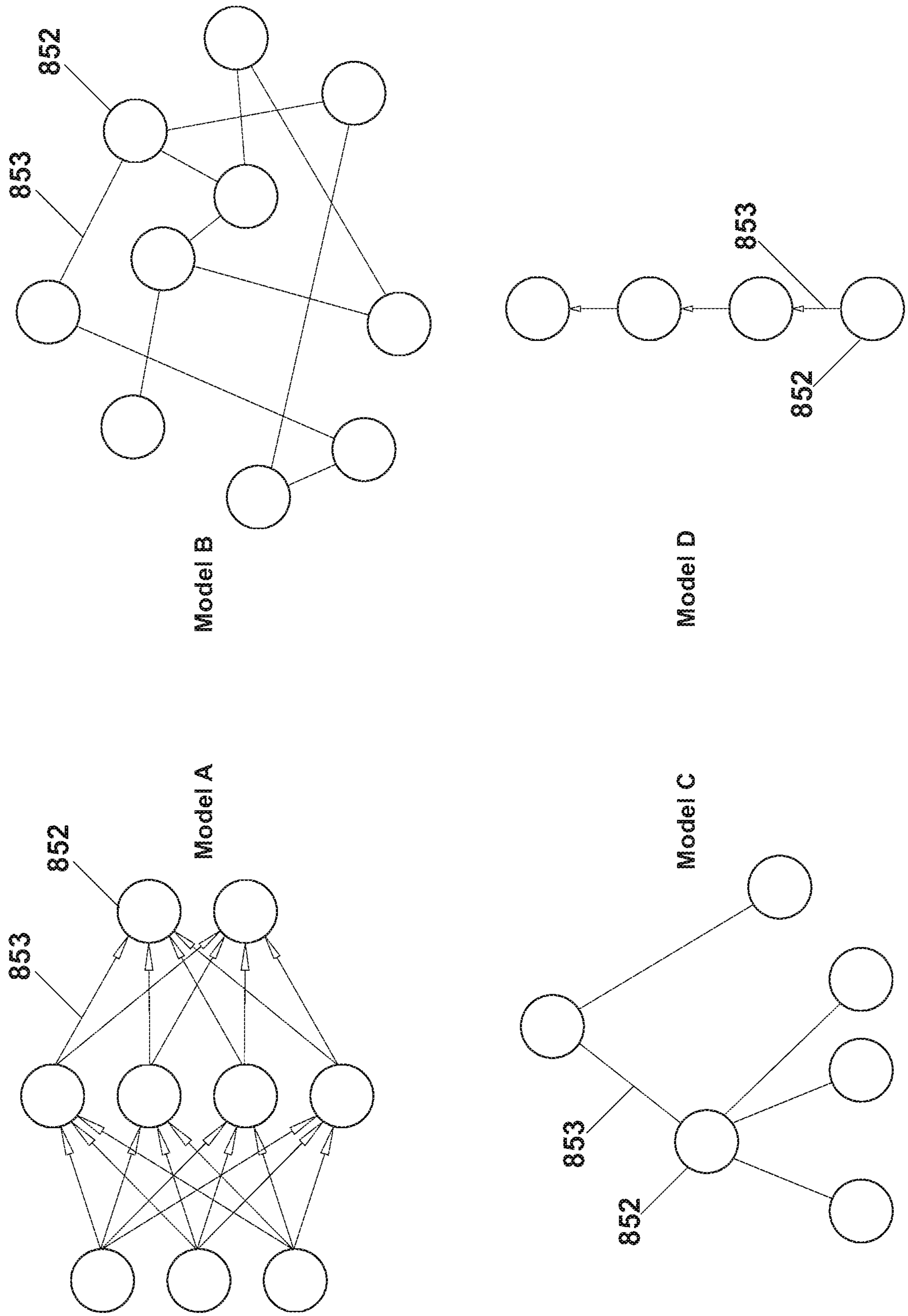


FIG. 17

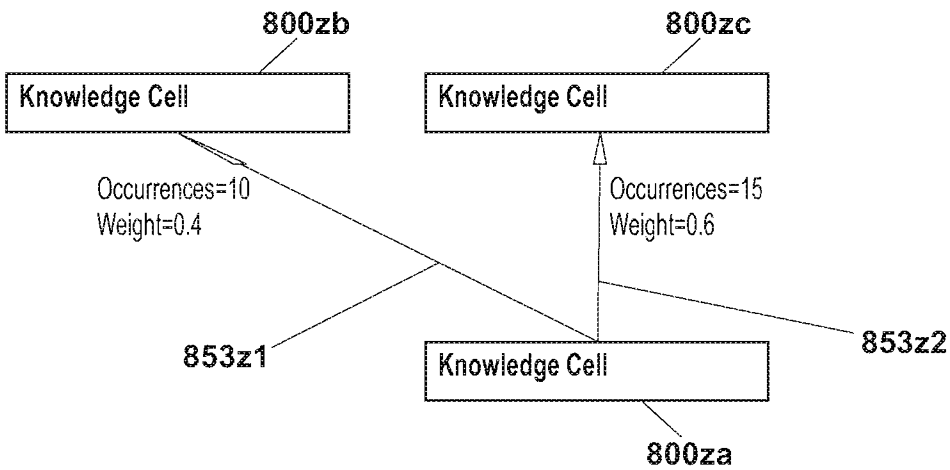


FIG. 18A

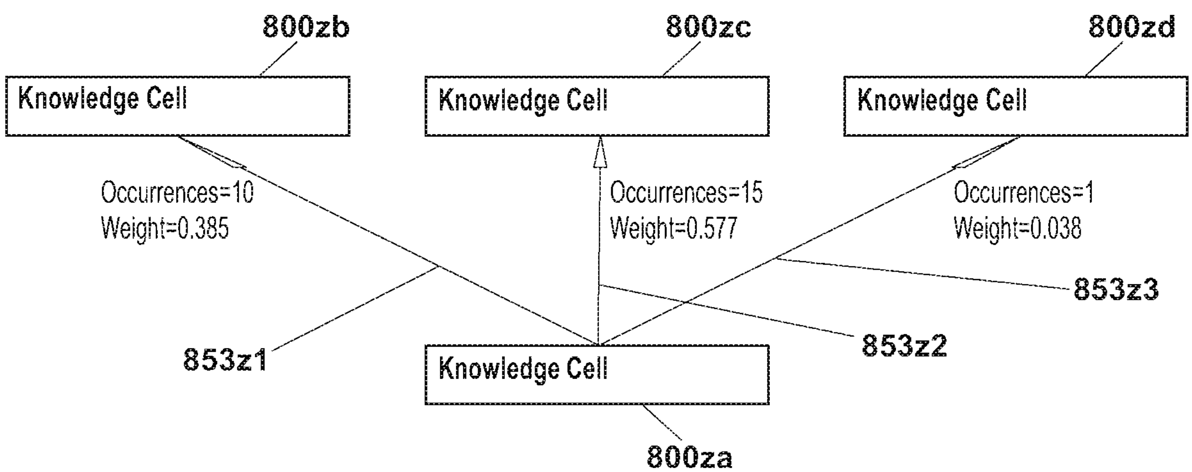


FIG. 18B

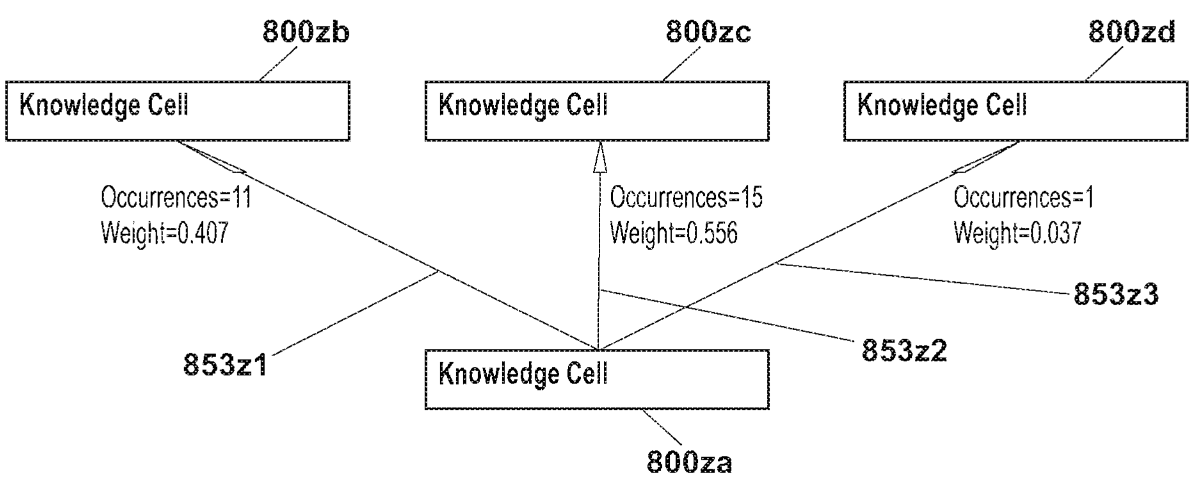


FIG. 18C

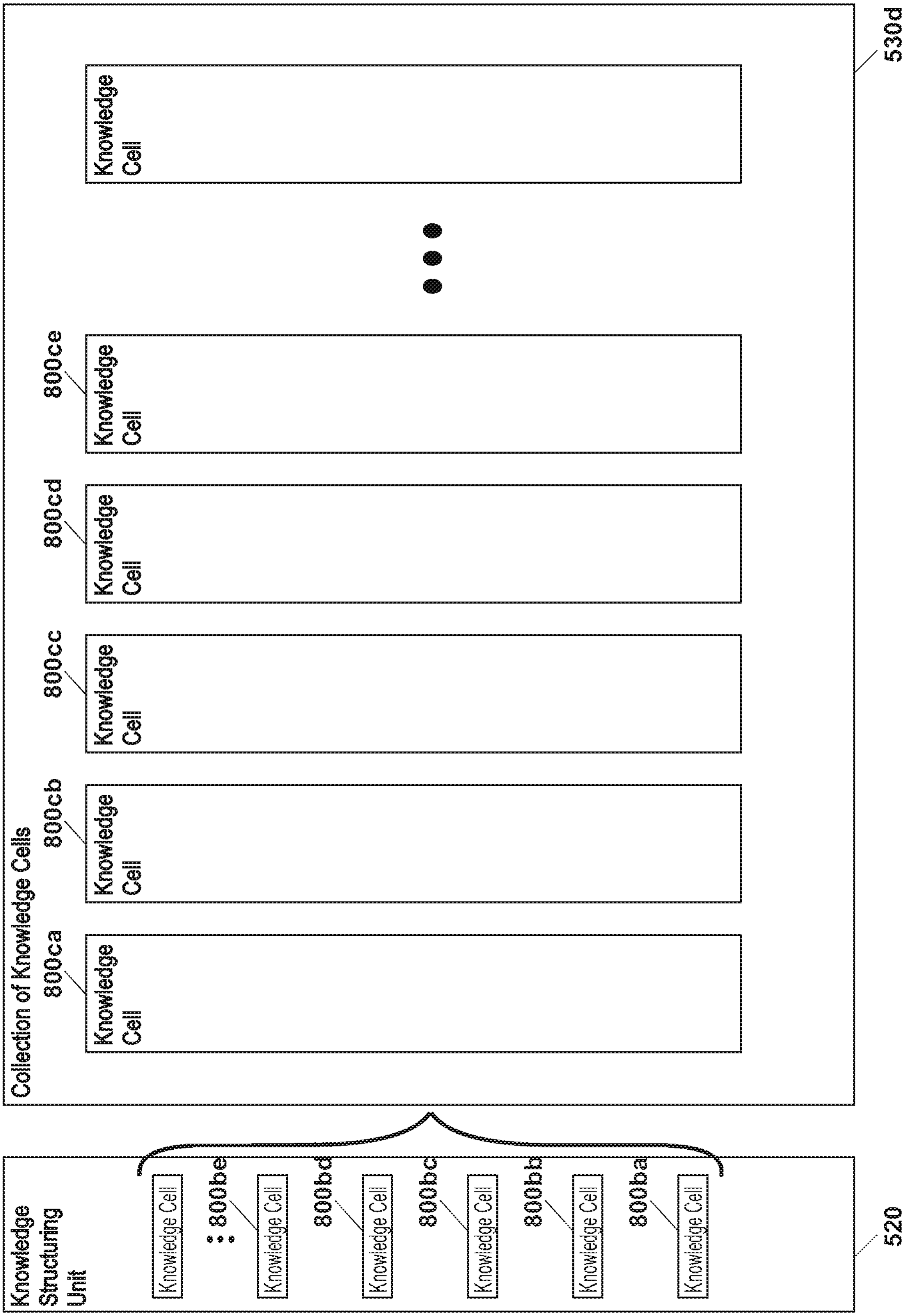
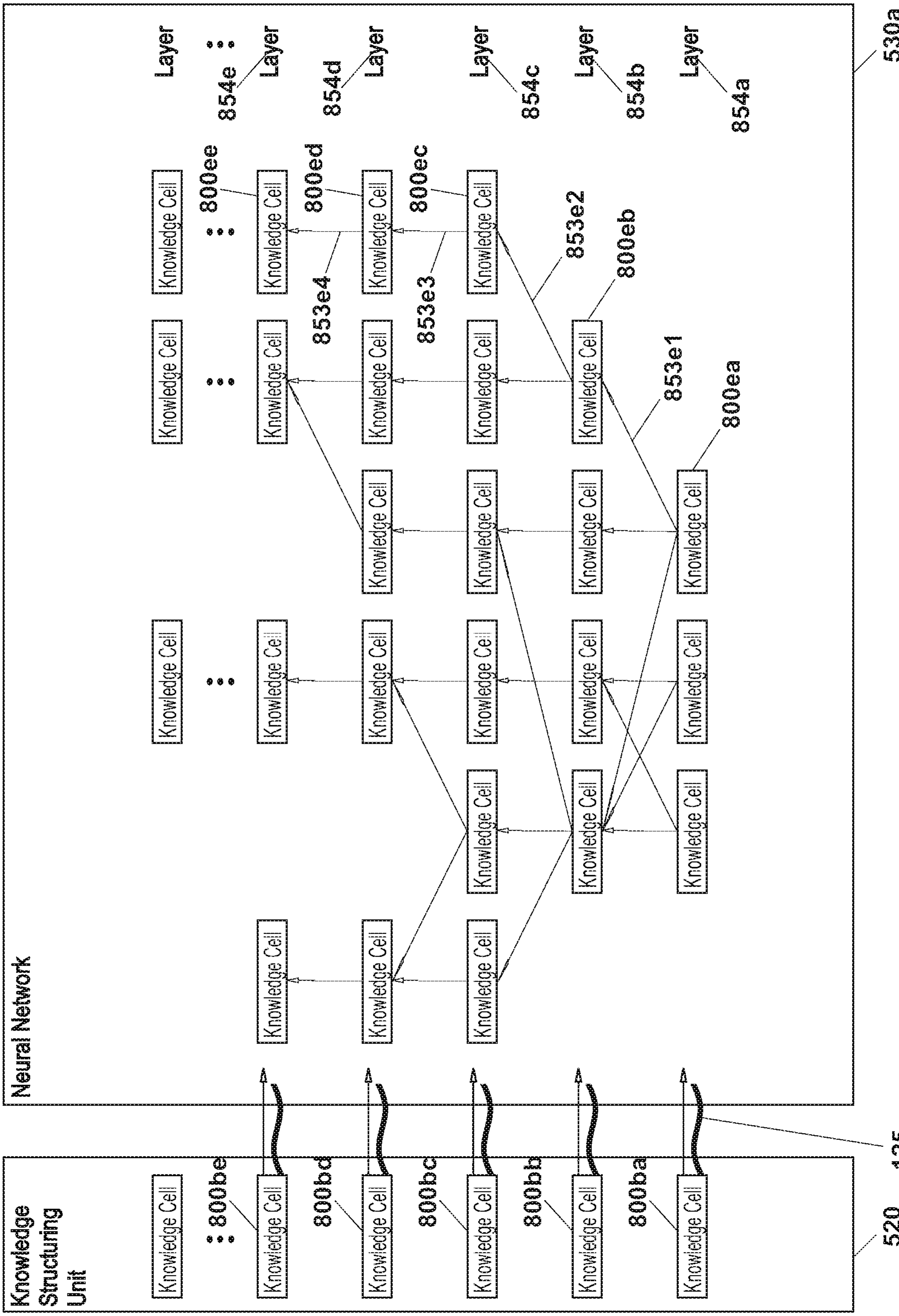
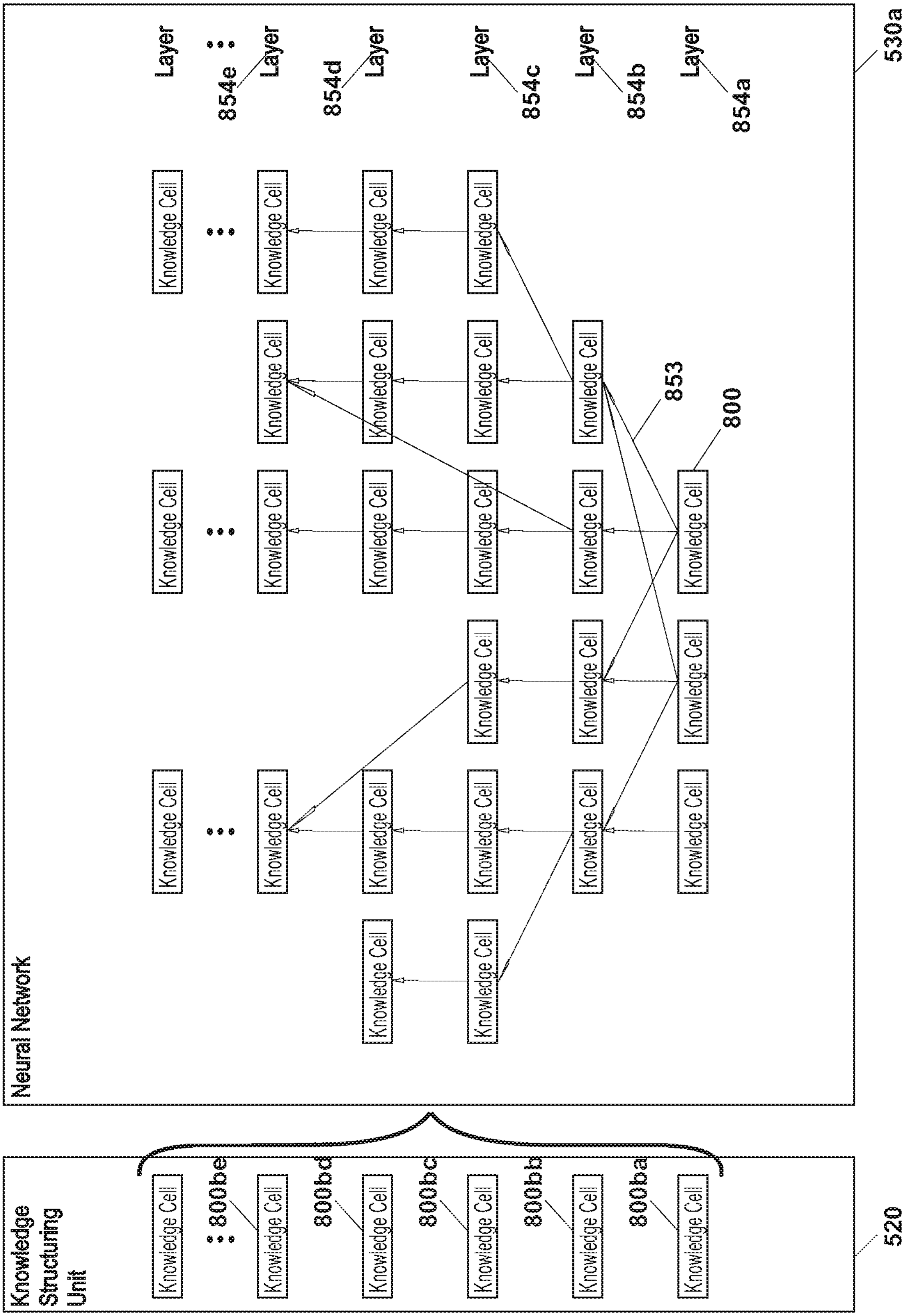

Collection of Knowledge Cells
Knowledge Cell
800ca
800cb
800cc
800cd
800ce
530d
Knowledge Structuring Unit
Knowledge Cell
800ba
800bb
800bc
800bd
800be
520

FIG. 19



Knowledge Structuring Unit
Neural Network
Knowledge Cell
Layer
854a
854b
854c
854d
854e
800ea
800eb
800ec
800ed
800ee
853e1
853e2
853e3
853e4
800ba
800bb
800bc
800bd
800be
530a
520
125

FIG. 20



Knowledge Structuring Unit
Knowledge Cell
800be
800bd
800bc
800bb
800ba
520
Neural Network
Layer
854e
854d
854c
854b
854a
853
800
530a

FIG. 21

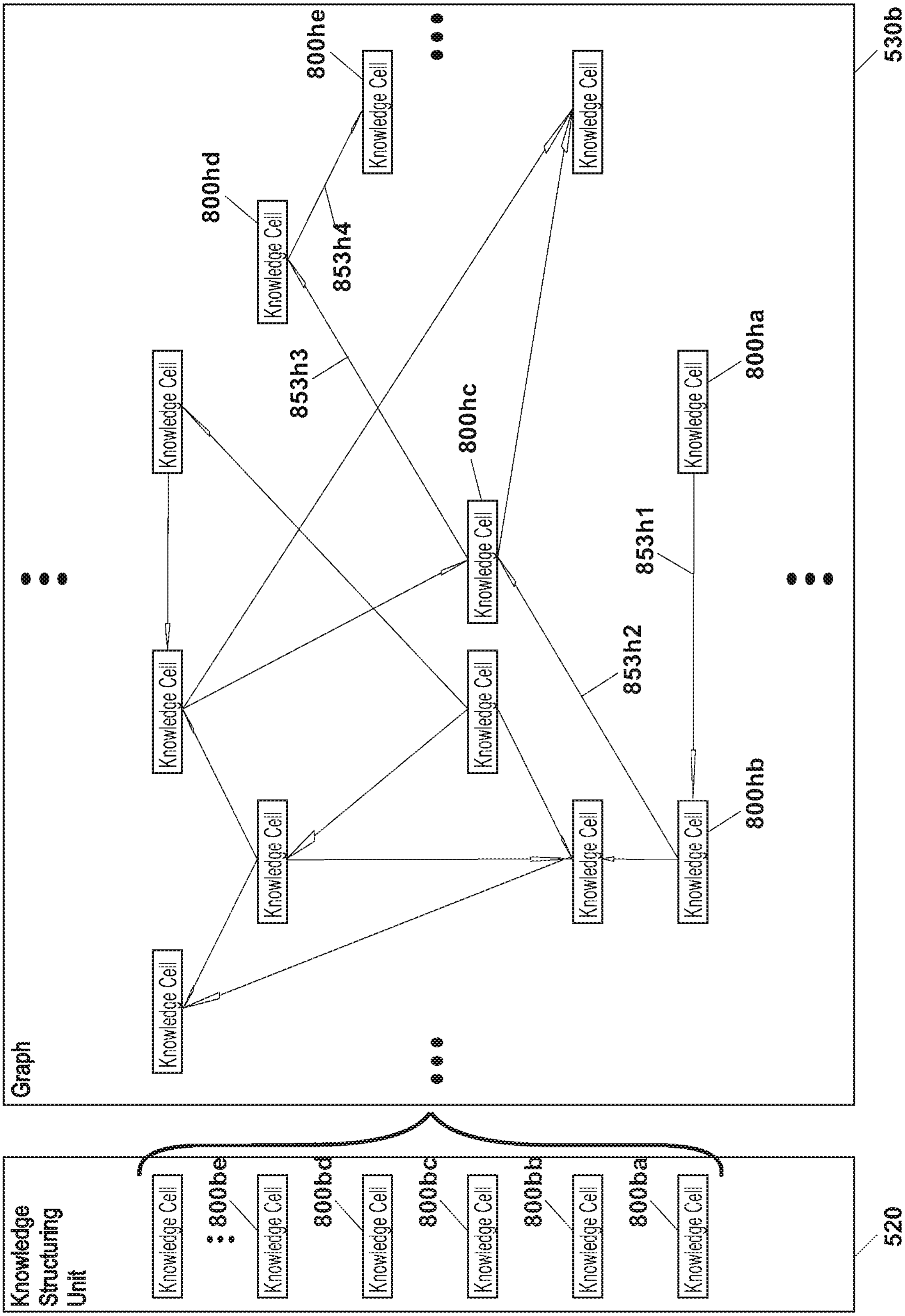

Knowledge Structuring Unit
Knowledge Cell
800ba
800bb
800bc
800bd
800be
520
Graph
800ha
800hb
800hc
800hd
800he
853h1
853h2
853h3
853h4
530b

FIG. 22

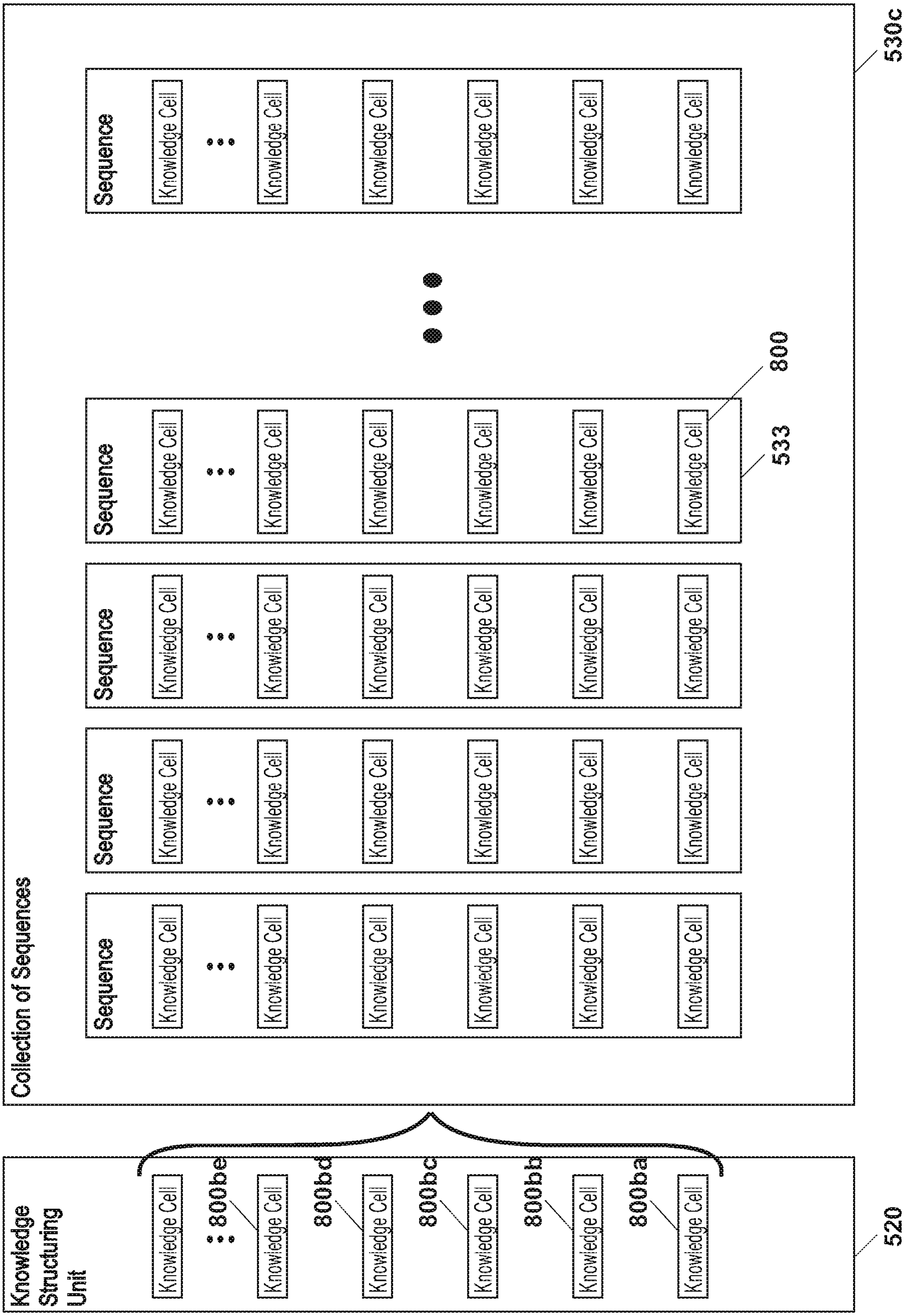

Knowledge Structuring Unit
Knowledge Cell
800ba
800bb
800bc
800bd
800be
520
Collection of Sequences
Sequence
Knowledge Cell
533
800
530c

FIG. 23

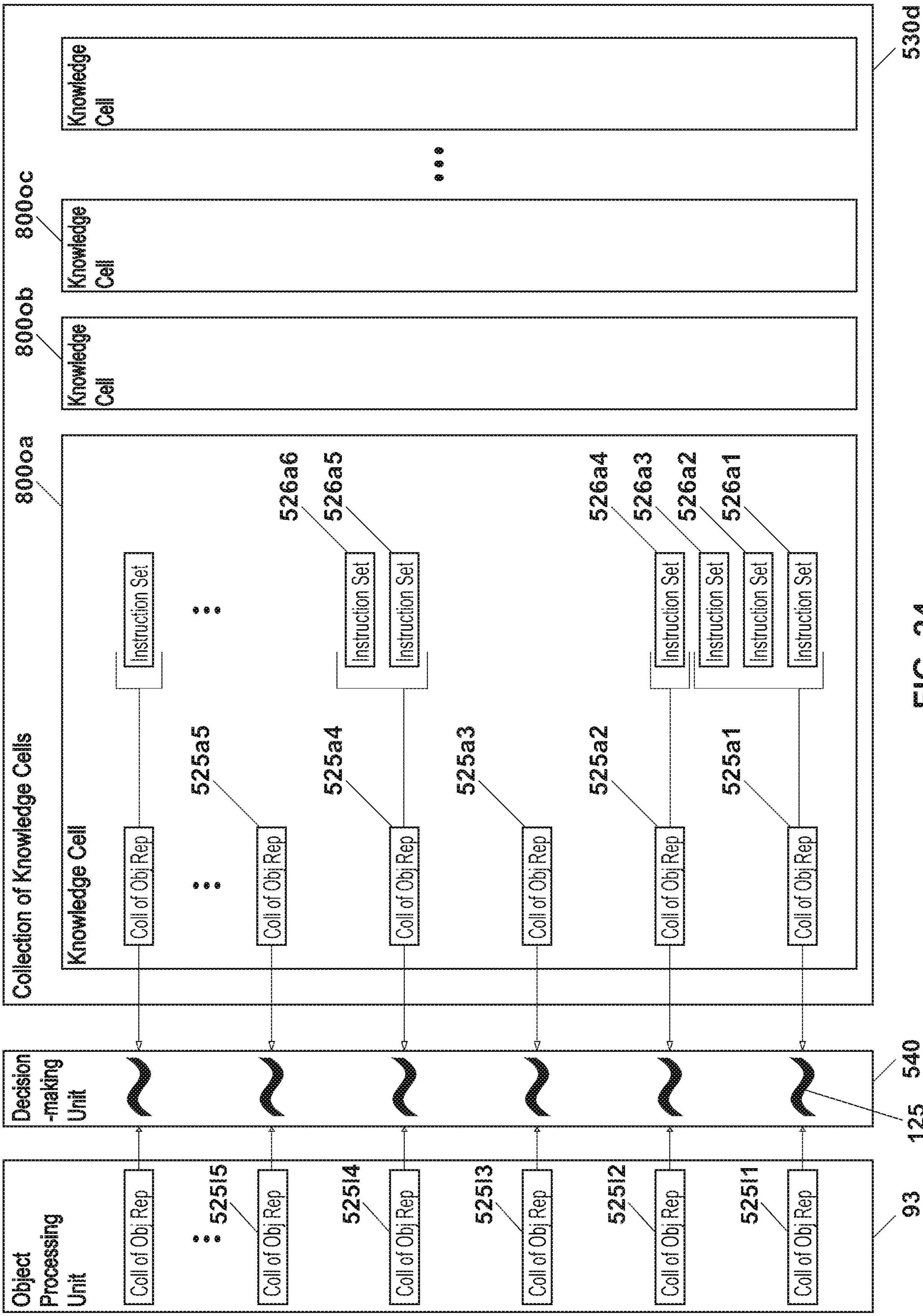

Object Processing Unit
Coll of Obj Rep
525l1
525l2
525l3
525l4
525l5
93
Decision -making Unit
125
540
Collection of Knowledge Cells
Knowledge Cell
525a1
525a2
525a3
525a4
525a5
Instruction Set
526a1
526a2
526a3
526a4
526a5
526a6
800oa
800ob
800oc
530d

FIG. 24

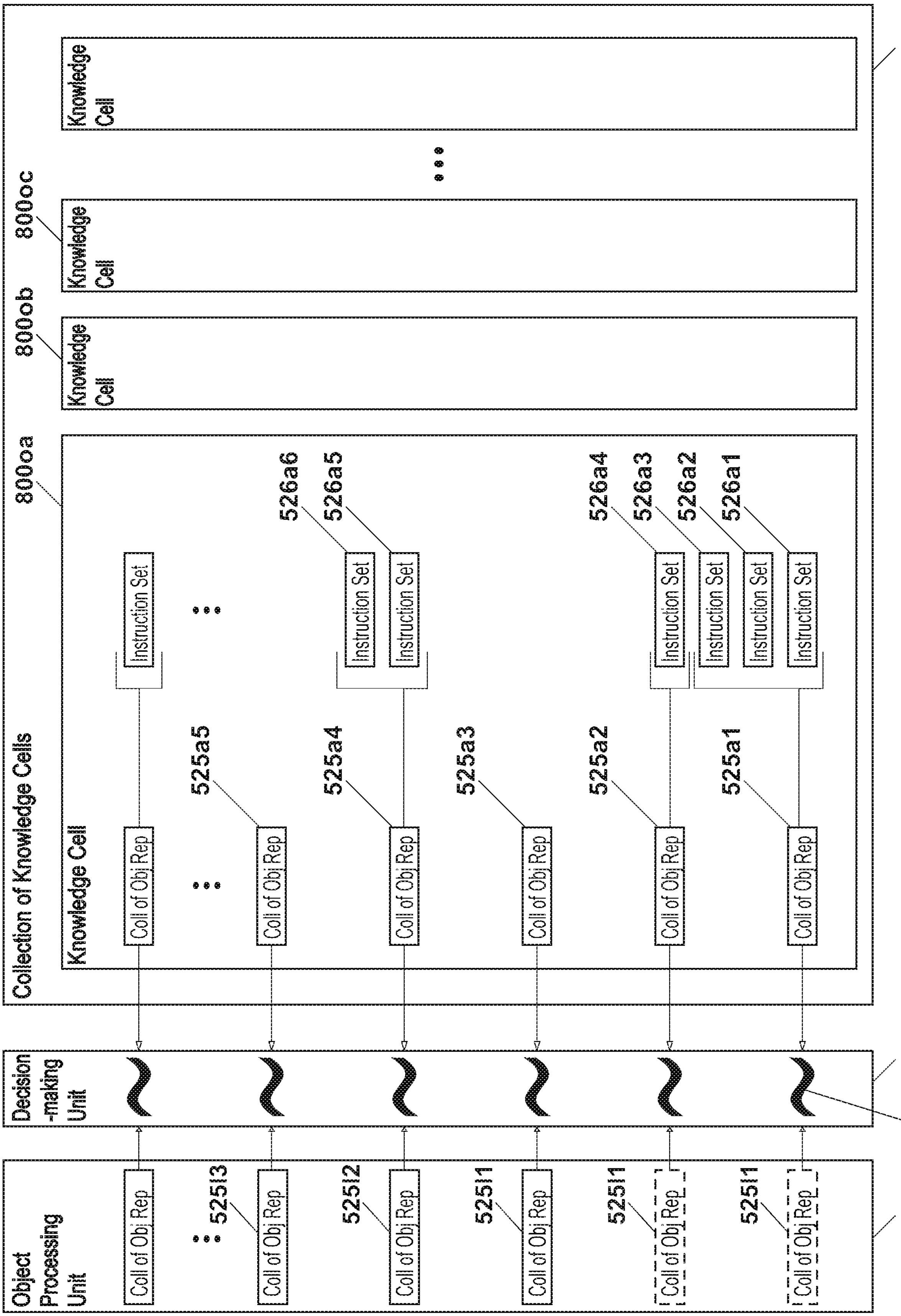

Object Processing Unit
Coll of Obj Rep
525l3
525l2
525l1
525l1
525l1
93
Decision -making Unit
125
540
Collection of Knowledge Cells
Knowledge Cell
Coll of Obj Rep
Instruction Set
525a5
525a4
525a3
525a2
525a1
526a6
526a5
526a4
526a3
526a2
526a1
800oa
800ob
800oc
Knowledge Cell
530d

FIG. 25

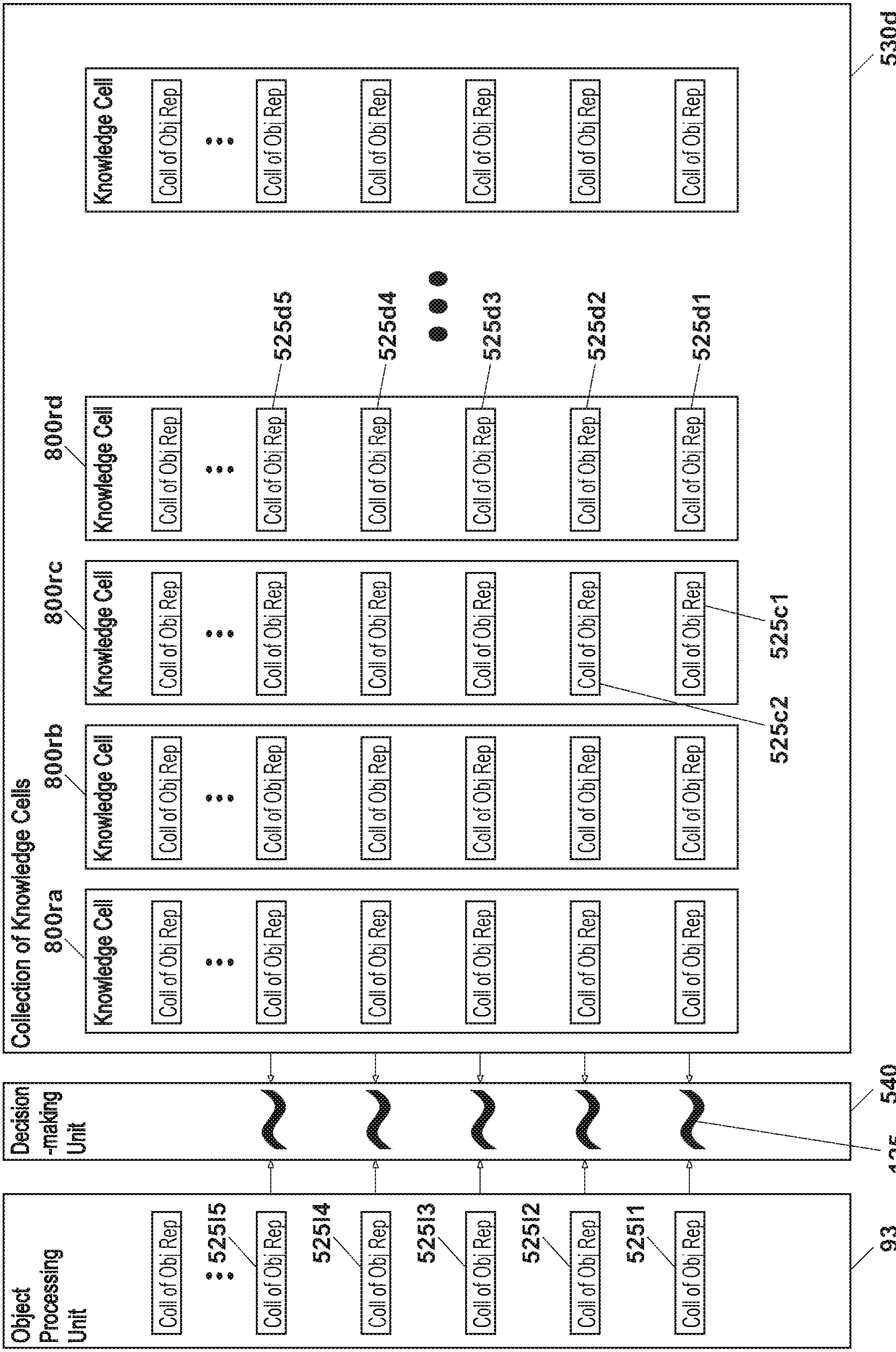
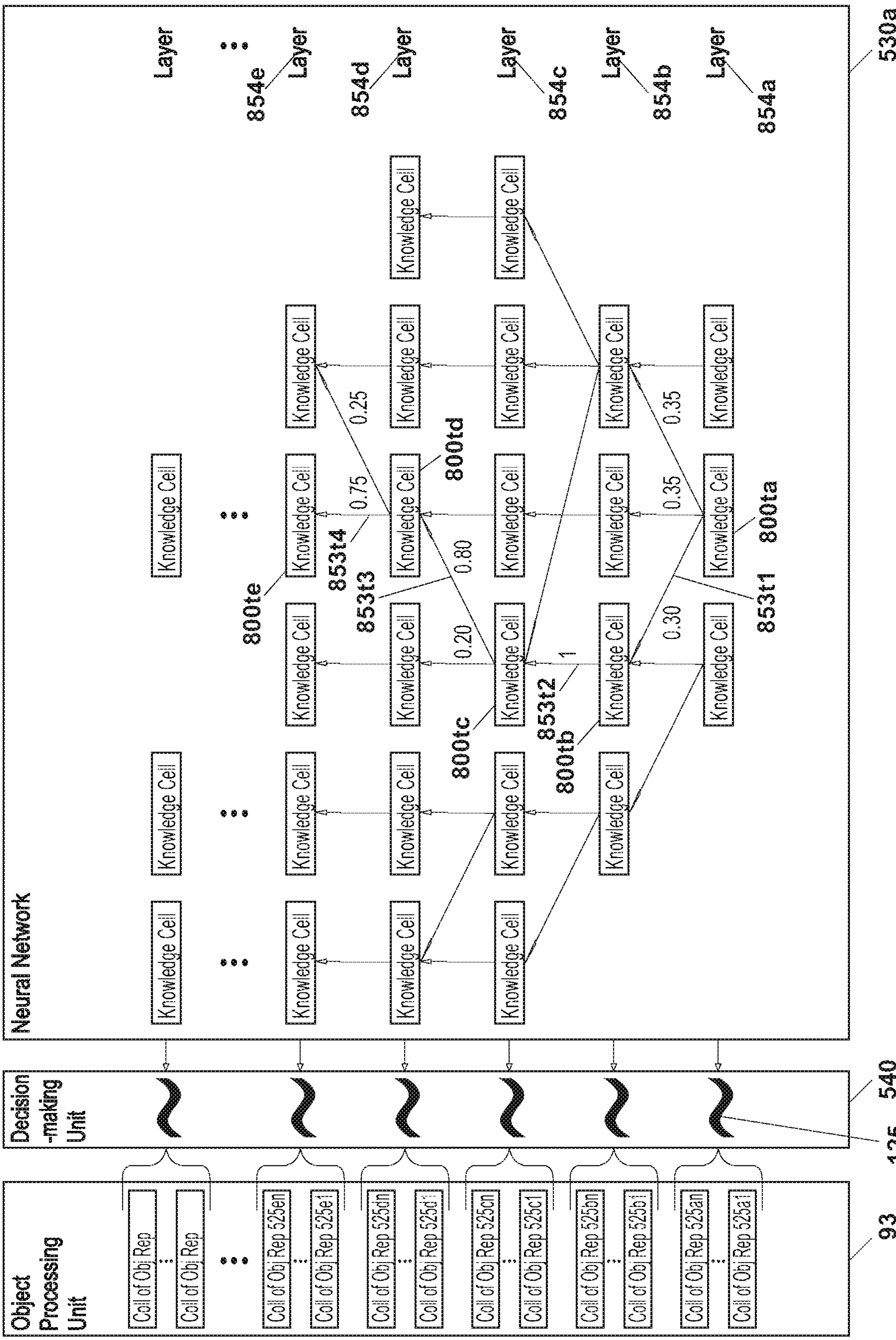
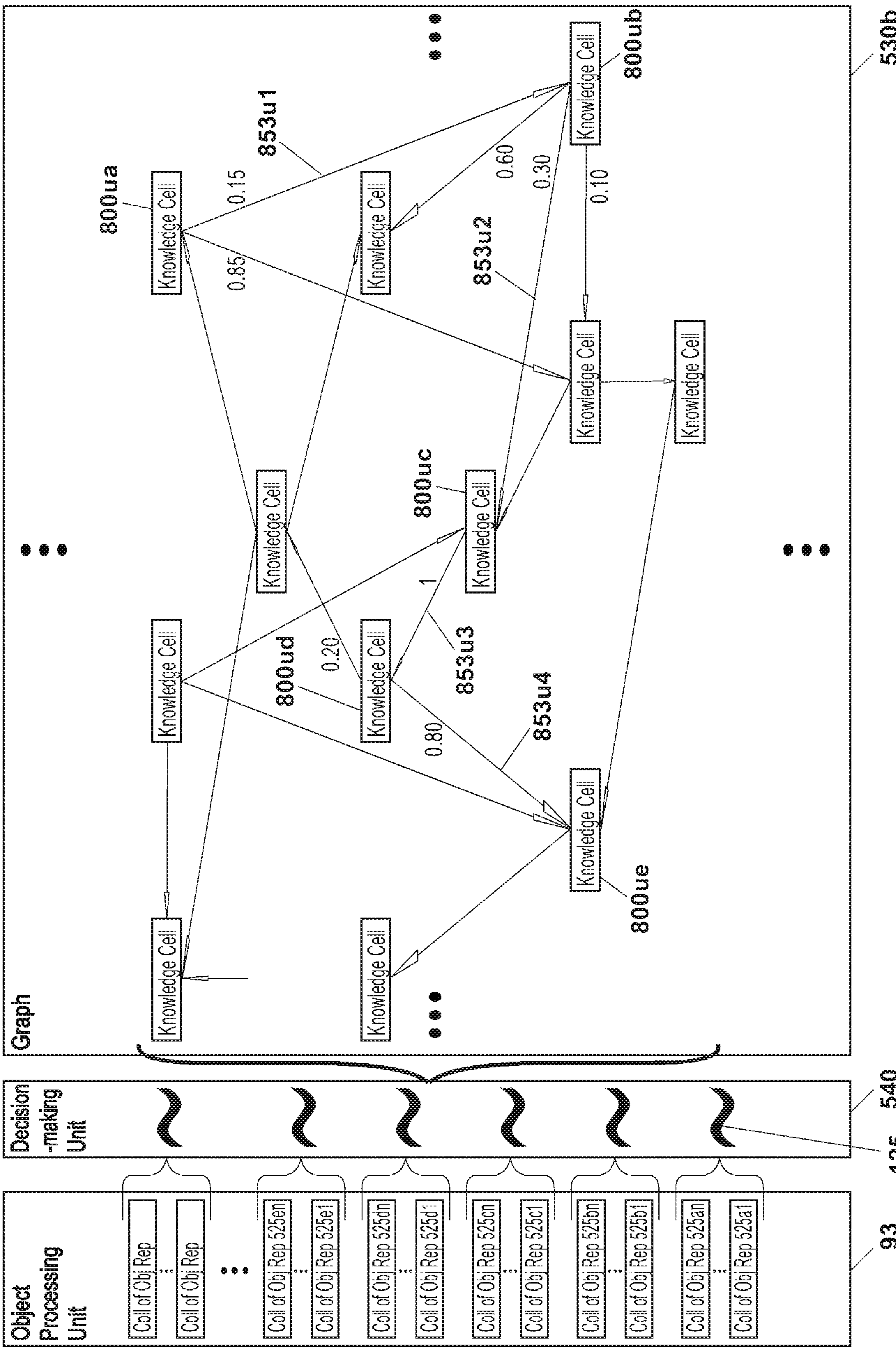

Object Processing Unit
Coll of Obj Rep
525l5
525l4
525l3
525l2
525l1
93
Decision -making Unit
125
540
Collection of Knowledge Cells
Knowledge Cell
800ra
800rb
800rc
800rd
525c1
525c2
525d1
525d2
525d3
525d4
525d5
530d

FIG. 26



Neural Network
Knowledge Cell
Layer
854a
854b
854c
854d
854e
800ta
800tb
800tc
800td
800te
853t1
853t2
853t3
853t4
0.30
1
0.20
0.80
0.75
0.35
0.35
0.25
Decision -making Unit
Object Processing Unit
Coll of Obj Rep 525an
Coll of Obj Rep 525a1
Coll of Obj Rep 525bn
Coll of Obj Rep 525b1
Coll of Obj Rep 525cn
Coll of Obj Rep 525c1
Coll of Obj Rep 525dn
Coll of Obj Rep 525d1
Coll of Obj Rep 525en
Coll of Obj Rep 525e1
Coll of Obj Rep
530a
540
125
93

FIG. 27



Object Processing Unit
Coll of Obj Rep
Coll of Obj Rep
Coll of Obj Rep 525en
Coll of Obj Rep 525e1
Coll of Obj Rep 525dn
Coll of Obj Rep 525d1
Coll of Obj Rep 525cn
Coll of Obj Rep 525c1
Coll of Obj Rep 525bn
Coll of Obj Rep 525b1
Coll of Obj Rep 525an
Coll of Obj Rep 525a1
93
Decision -making Unit
540
125
Graph
530b
Knowledge Cell
800ua
800ub
800uc
800ud
800ue
853u1
853u2
853u3
853u4
0.15
0.85
0.60
0.30
0.10
0.20
0.80
1

FIG. 28

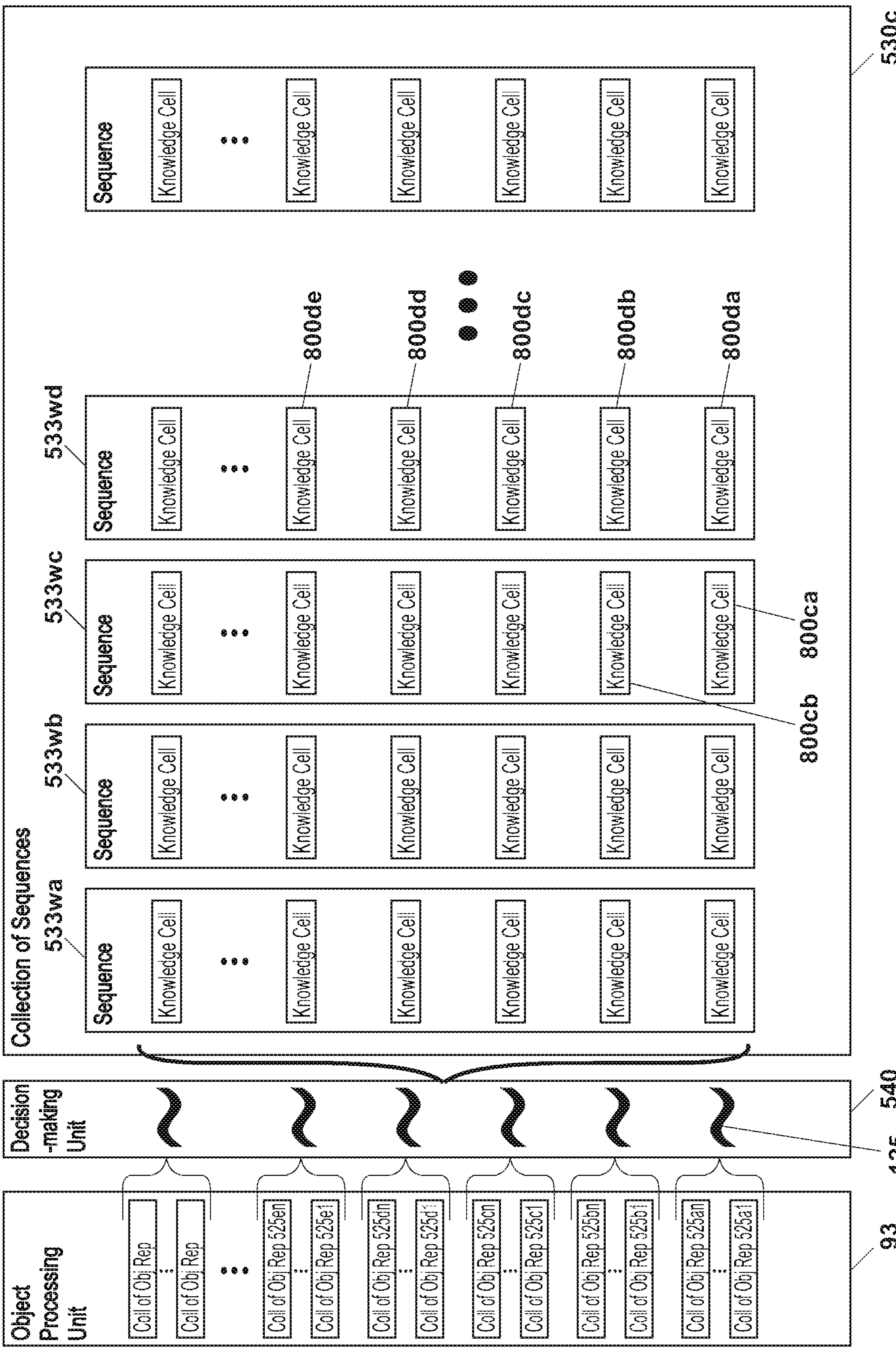

Object Processing Unit
Coll of Obj Rep
Coll of Obj Rep 525en
Coll of Obj Rep 525e1
Coll of Obj Rep 525dn
Coll of Obj Rep 525cn
Coll of Obj Rep 525c1
Coll of Obj Rep 525bn
Coll of Obj Rep 525b1
Coll of Obj Rep 525an
Coll of Obj Rep 525a1
93
Decision -making Unit
540
125
Collection of Sequences
Sequence
Knowledge Cell
533wa
533wb
533wc
533wd
800ca
800cb
800da
800db
800dc
800dd
800de
530c

FIG. 29

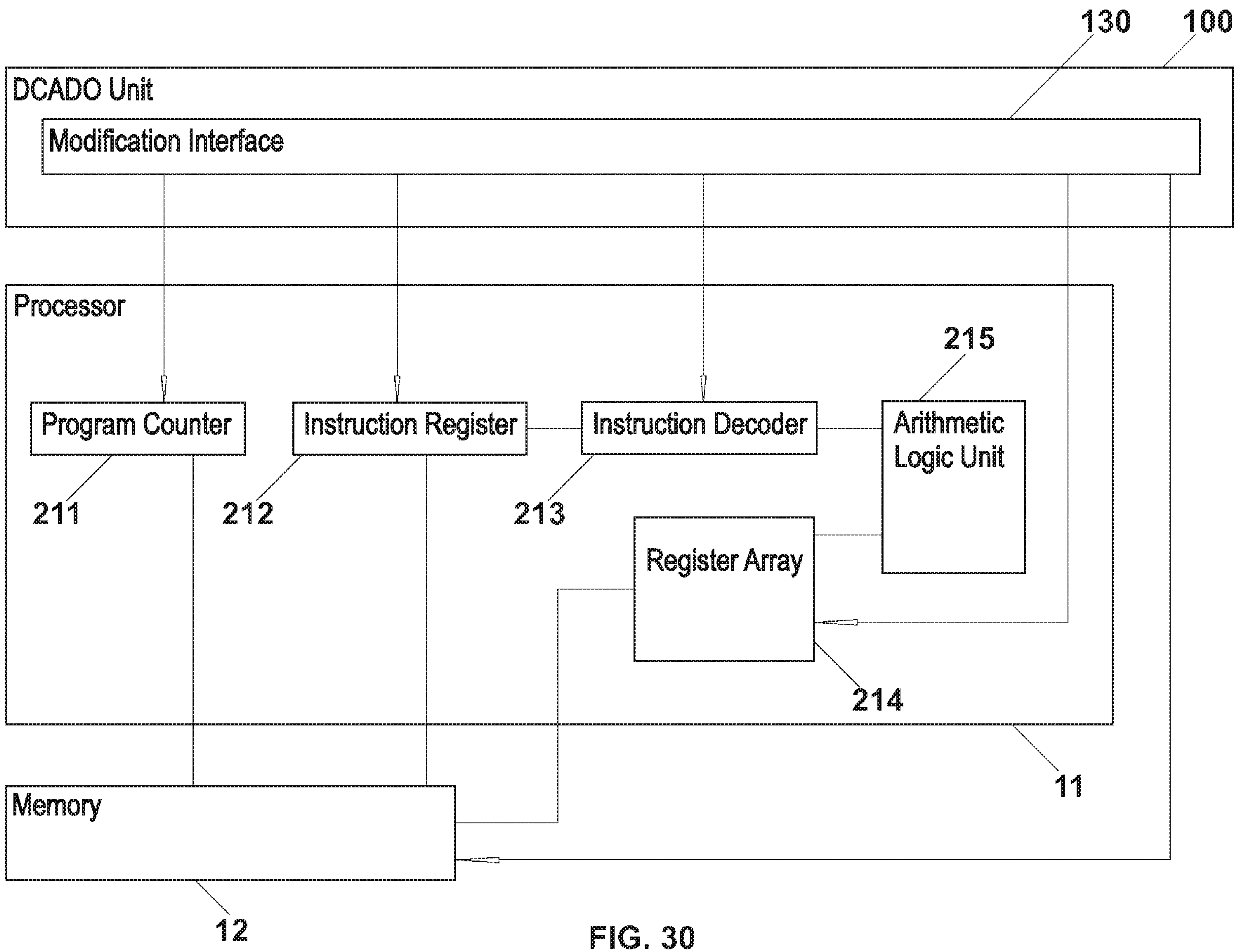

130
100
DCADO Unit
Modification Interface
Processor
215
Program Counter
Instruction Register
Instruction Decoder
Arithmetic Logic Unit
211
212
213
Register Array
214
11
Memory
12

FIG. 30

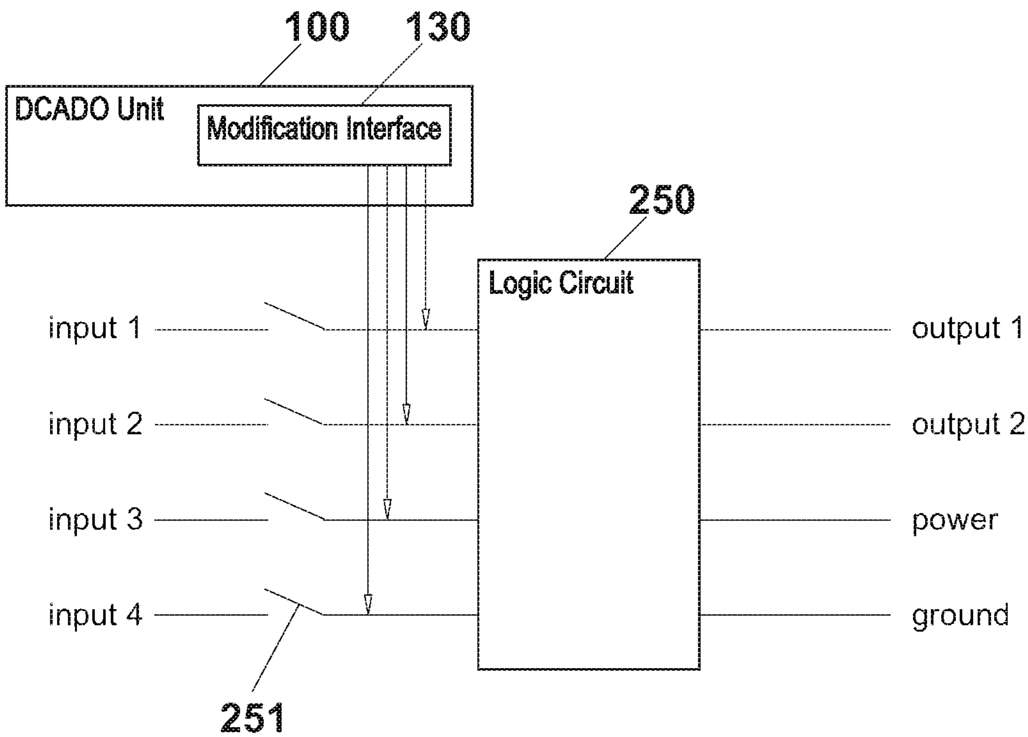


FIG. 31A

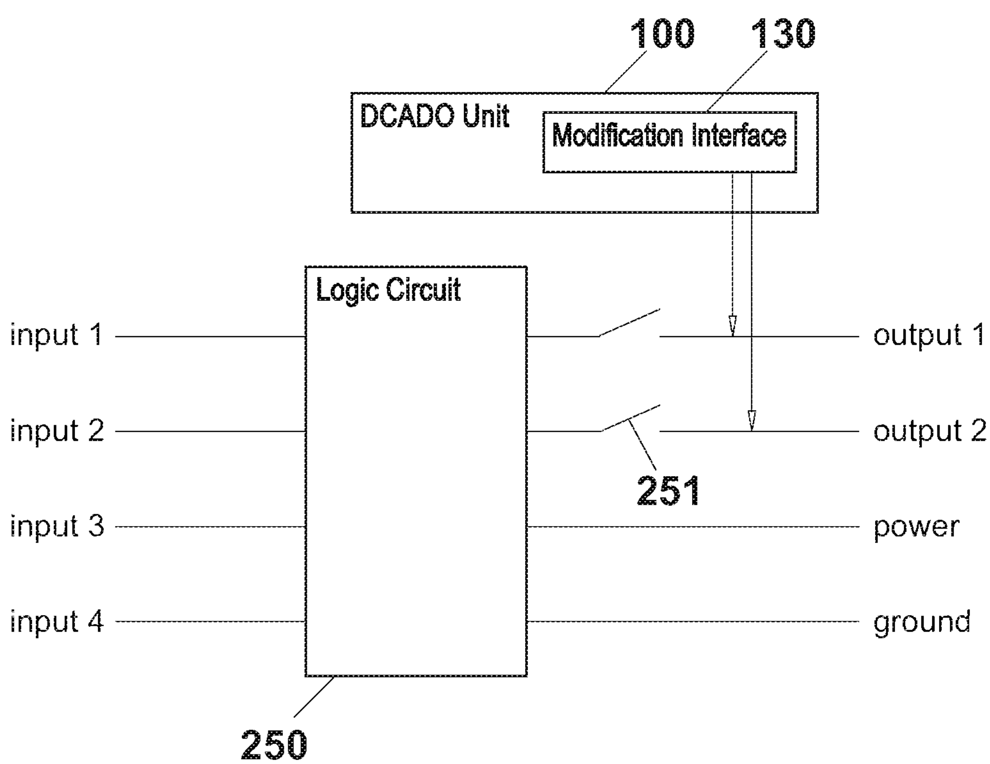


FIG. 31B

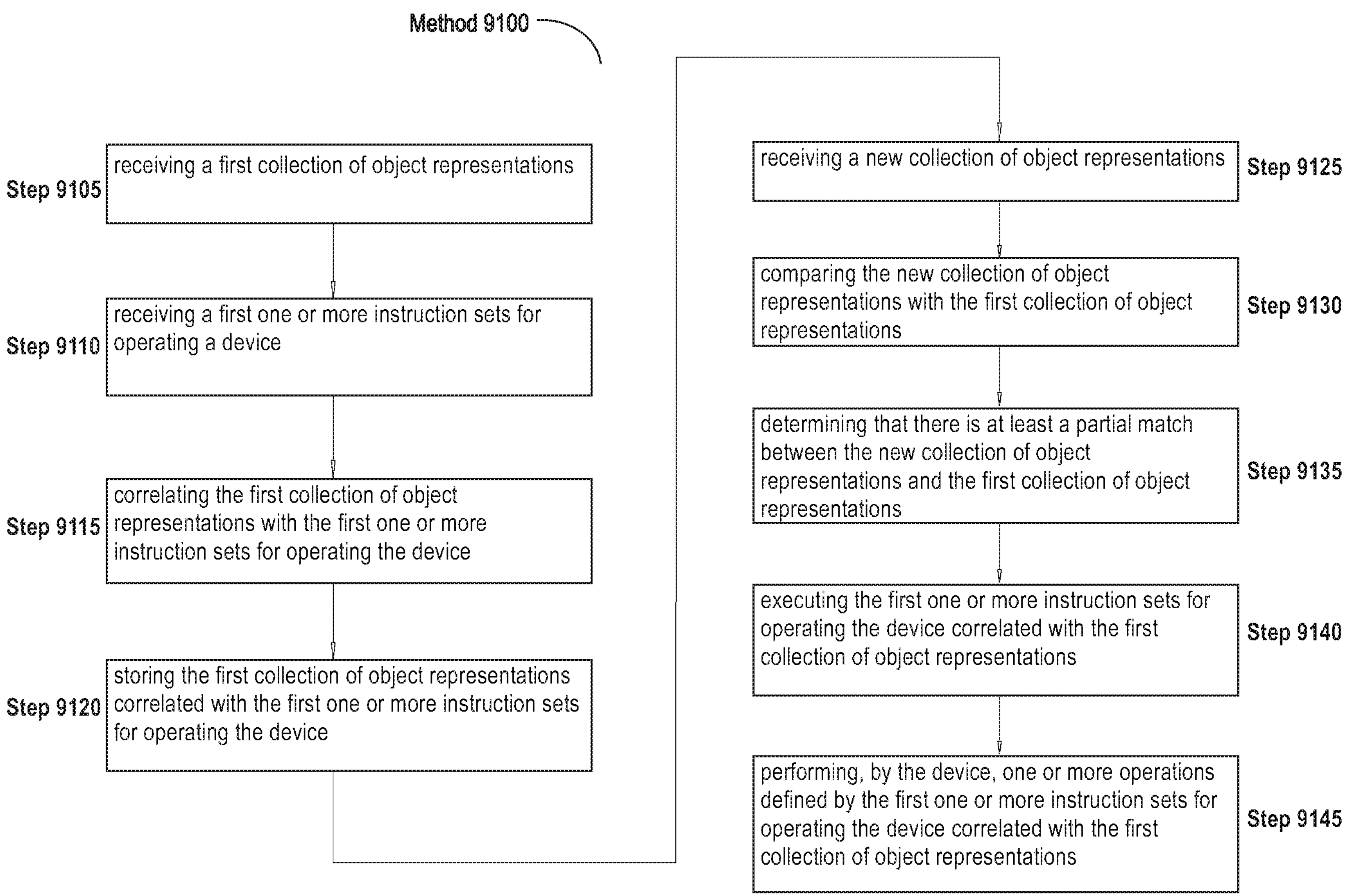

Method 9100
Step 9105
receiving a first collection of object representations
Step 9110
receiving a first one or more instruction sets for operating a device
Step 9115
correlating the first collection of object representations with the first one or more instruction sets for operating the device
Step 9120
storing the first collection of object representations correlated with the first one or more instruction sets for operating the device
Step 9125
receiving a new collection of object representations
Step 9130
comparing the new collection of object representations with the first collection of object representations
Step 9135
determining that there is at least a partial match between the new collection of object representations and the first collection of object representations
Step 9140
executing the first one or more instruction sets for operating the device correlated with the first collection of object representations
Step 9145
performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations

FIG. 32

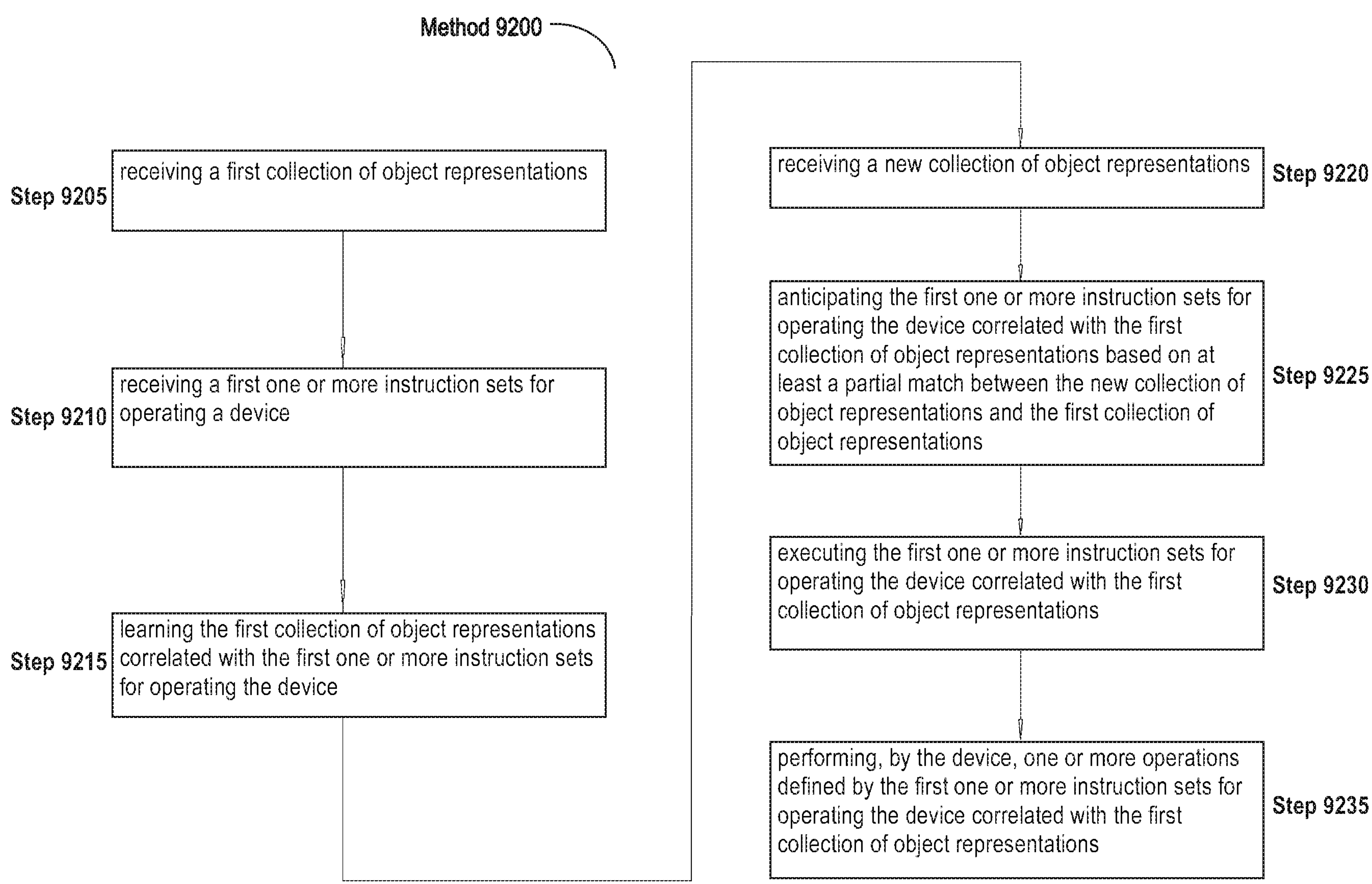

Method 9200
Step 9205
receiving a first collection of object representations
Step 9210
receiving a first one or more instruction sets for operating a device
Step 9215
learning the first collection of object representations correlated with the first one or more instruction sets for operating the device
Step 9220
receiving a new collection of object representations
Step 9225
anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations
Step 9230
executing the first one or more instruction sets for operating the device correlated with the first collection of object representations
Step 9235
performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations

FIG. 33

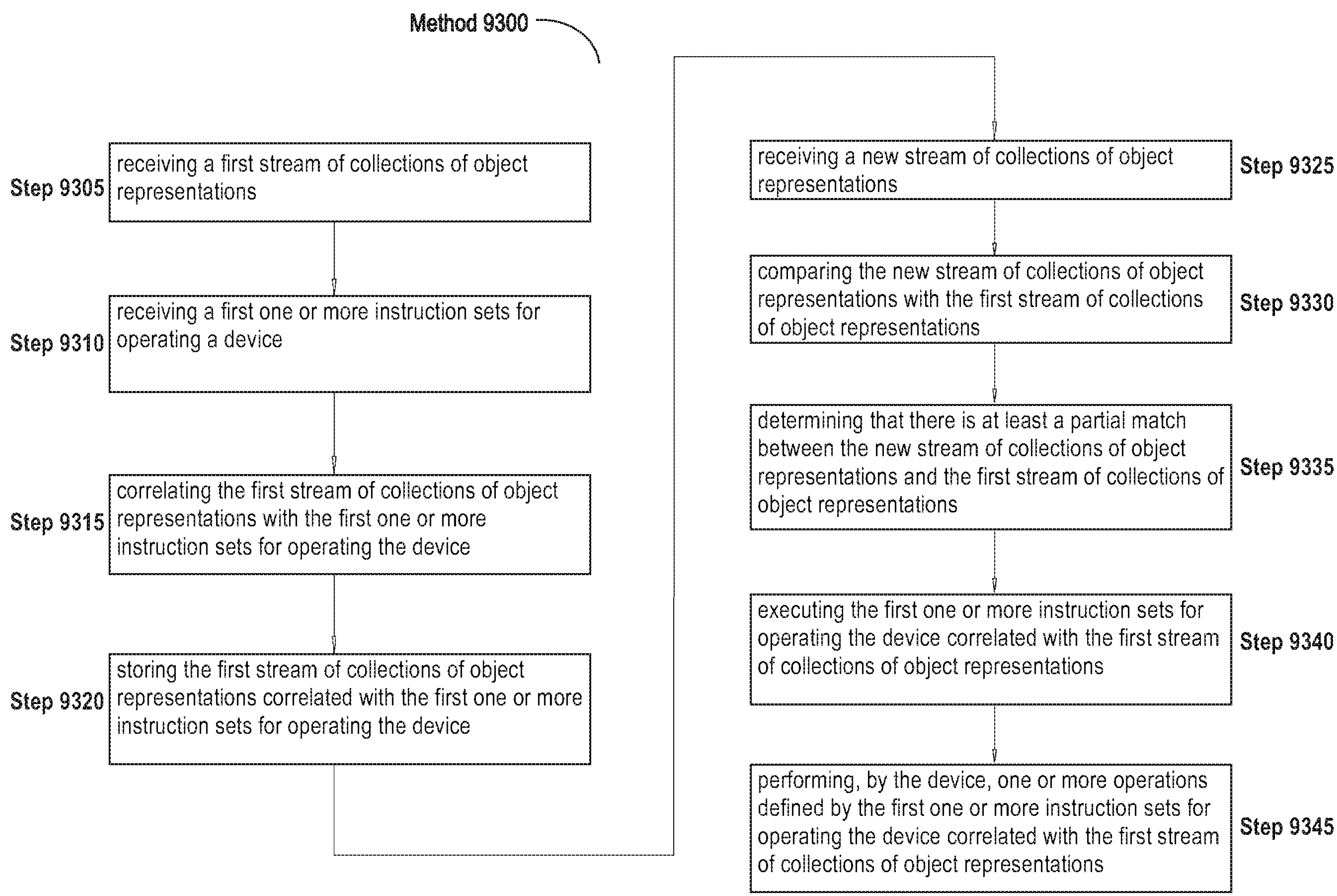

Method 9300
Step 9305
receiving a first stream of collections of object representations
Step 9310
receiving a first one or more instruction sets for operating a device
Step 9315
correlating the first stream of collections of object representations with the first one or more instruction sets for operating the device
Step 9320
storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device
Step 9325
receiving a new stream of collections of object representations
Step 9330
comparing the new stream of collections of object representations with the first stream of collections of object representations
Step 9335
determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations
Step 9340
executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations
Step 9345
performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations

FIG. 34

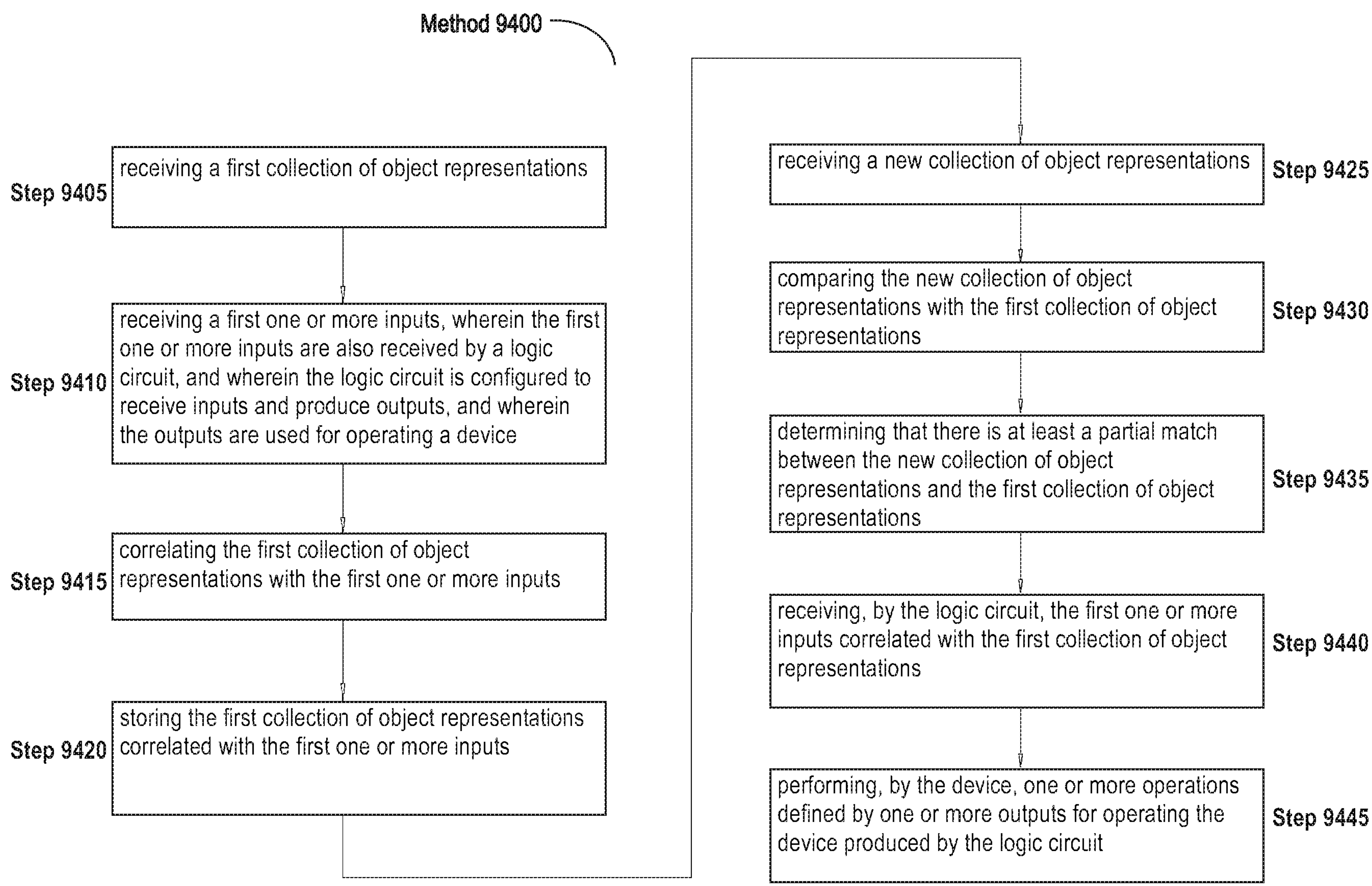

Method 9400
Step 9405
receiving a first collection of object representations
Step 9410
receiving a first one or more inputs, wherein the first one or more inputs are also received by a logic circuit, and wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device
Step 9415
correlating the first collection of object representations with the first one or more inputs
Step 9420
storing the first collection of object representations correlated with the first one or more inputs
Step 9425
receiving a new collection of object representations
Step 9430
comparing the new collection of object representations with the first collection of object representations
Step 9435
determining that there is at least a partial match between the new collection of object representations and the first collection of object representations
Step 9440
receiving, by the logic circuit, the first one or more inputs correlated with the first collection of object representations
Step 9445
performing, by the device, one or more operations defined by one or more outputs for operating the device produced by the logic circuit

FIG. 35

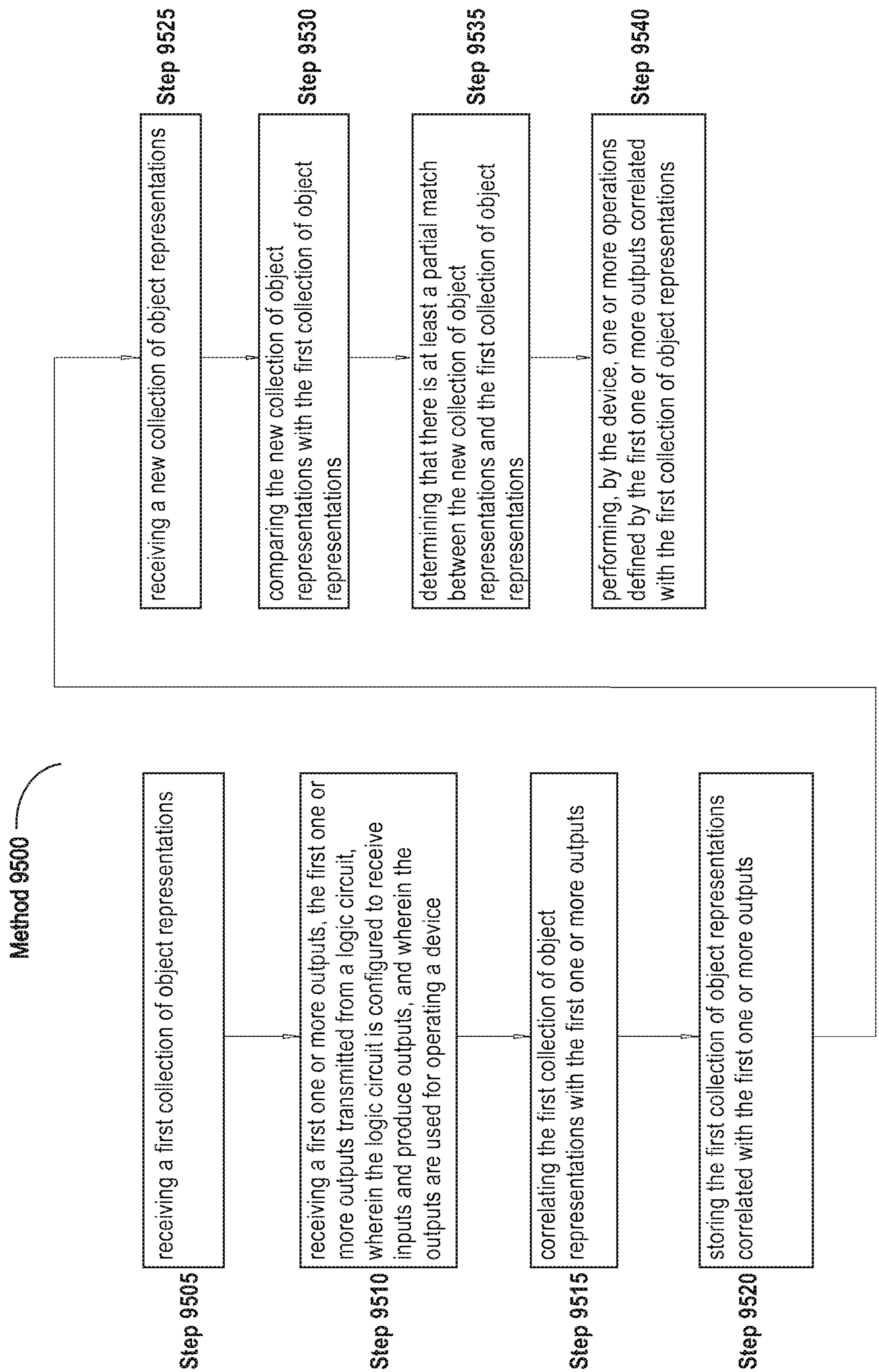

Method 9500
Step 9505
receiving a first collection of object representations
Step 9510
receiving a first one or more outputs, the first one or more outputs transmitted from a logic circuit, wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device
Step 9515
correlating the first collection of object representations with the first one or more outputs
Step 9520
storing the first collection of object representations correlated with the first one or more outputs
Step 9525
receiving a new collection of object representations
Step 9530
comparing the new collection of object representations with the first collection of object representations
Step 9535
determining that there is at least a partial match between the new collection of object representations and the first collection of object representations
Step 9540
performing, by the device, one or more operations defined by the first one or more outputs correlated with the first collection of object representations

FIG. 36

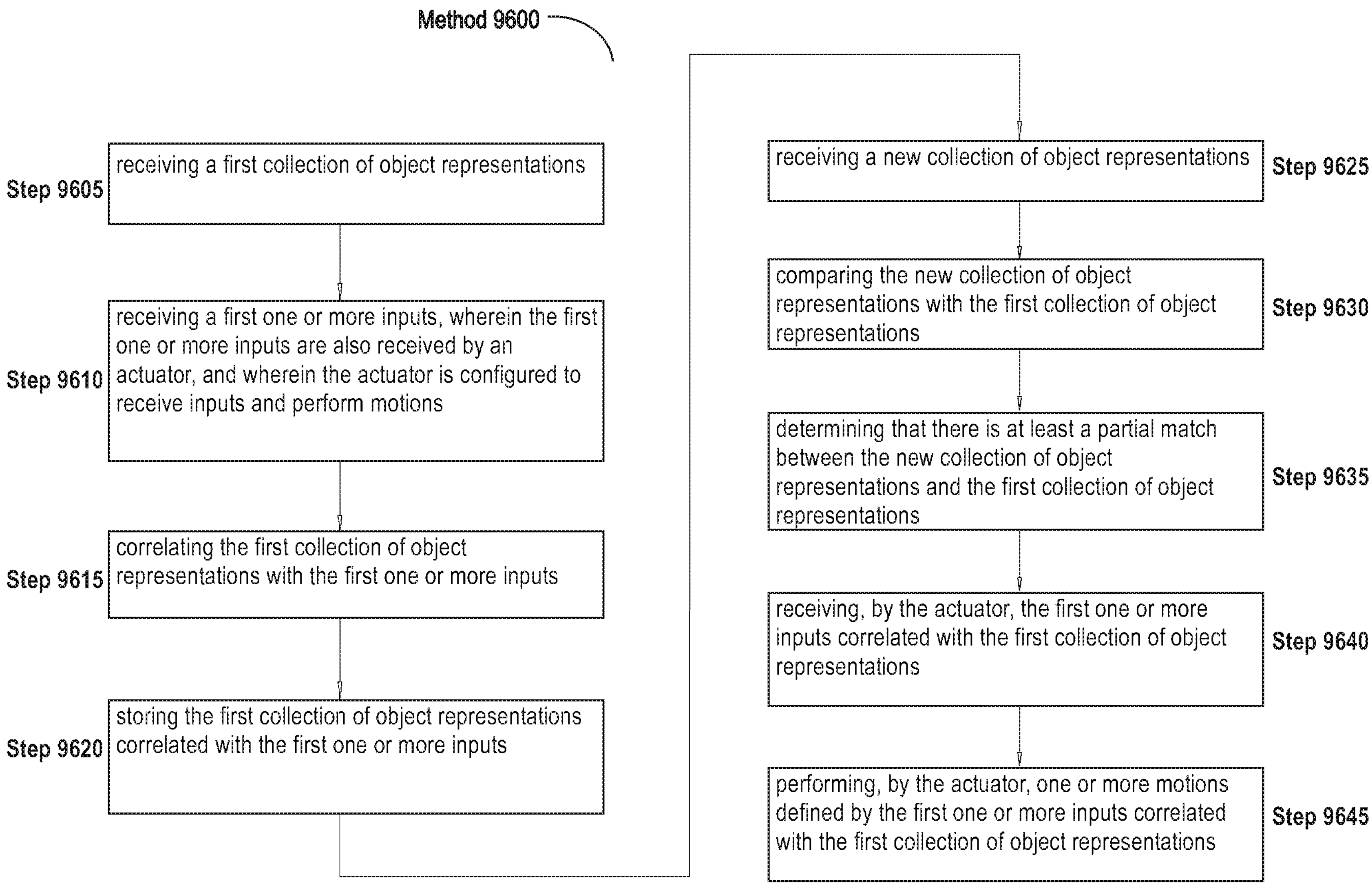

Method 9600
Step 9605
receiving a first collection of object representations
Step 9610
receiving a first one or more inputs, wherein the first one or more inputs are also received by an actuator, and wherein the actuator is configured to receive inputs and perform motions
Step 9615
correlating the first collection of object representations with the first one or more inputs
Step 9620
storing the first collection of object representations correlated with the first one or more inputs
Step 9625
receiving a new collection of object representations
Step 9630
comparing the new collection of object representations with the first collection of object representations
Step 9635
determining that there is at least a partial match between the new collection of object representations and the first collection of object representations
Step 9640
receiving, by the actuator, the first one or more inputs correlated with the first collection of object representations
Step 9645
performing, by the actuator, one or more motions defined by the first one or more inputs correlated with the first collection of object representations

FIG. 37

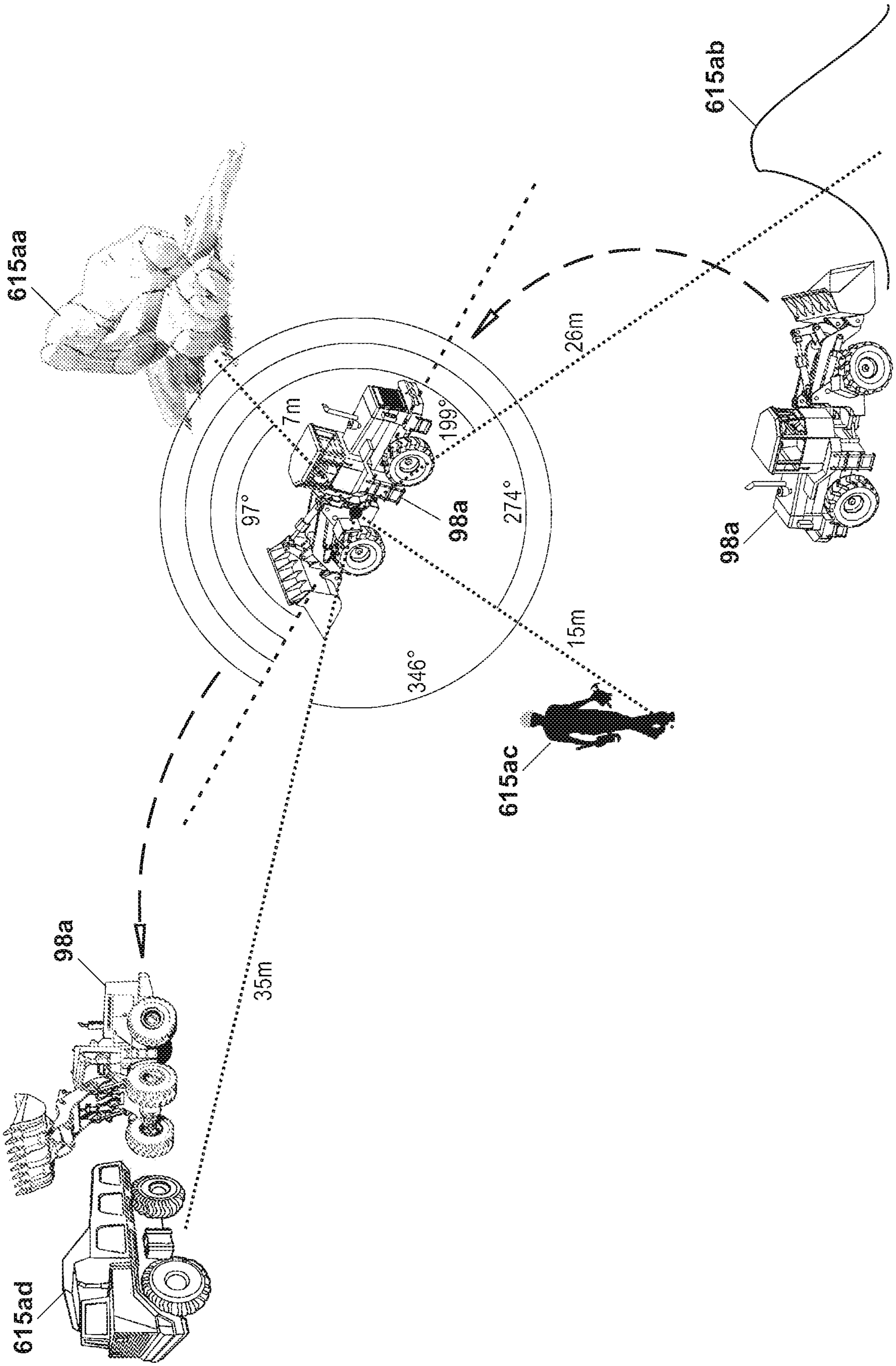

615aa
615ab
26m
7m
199°
97°
98a
274°
98a
15m
346°
615ac
98a
35m
615ad

FIG. 38

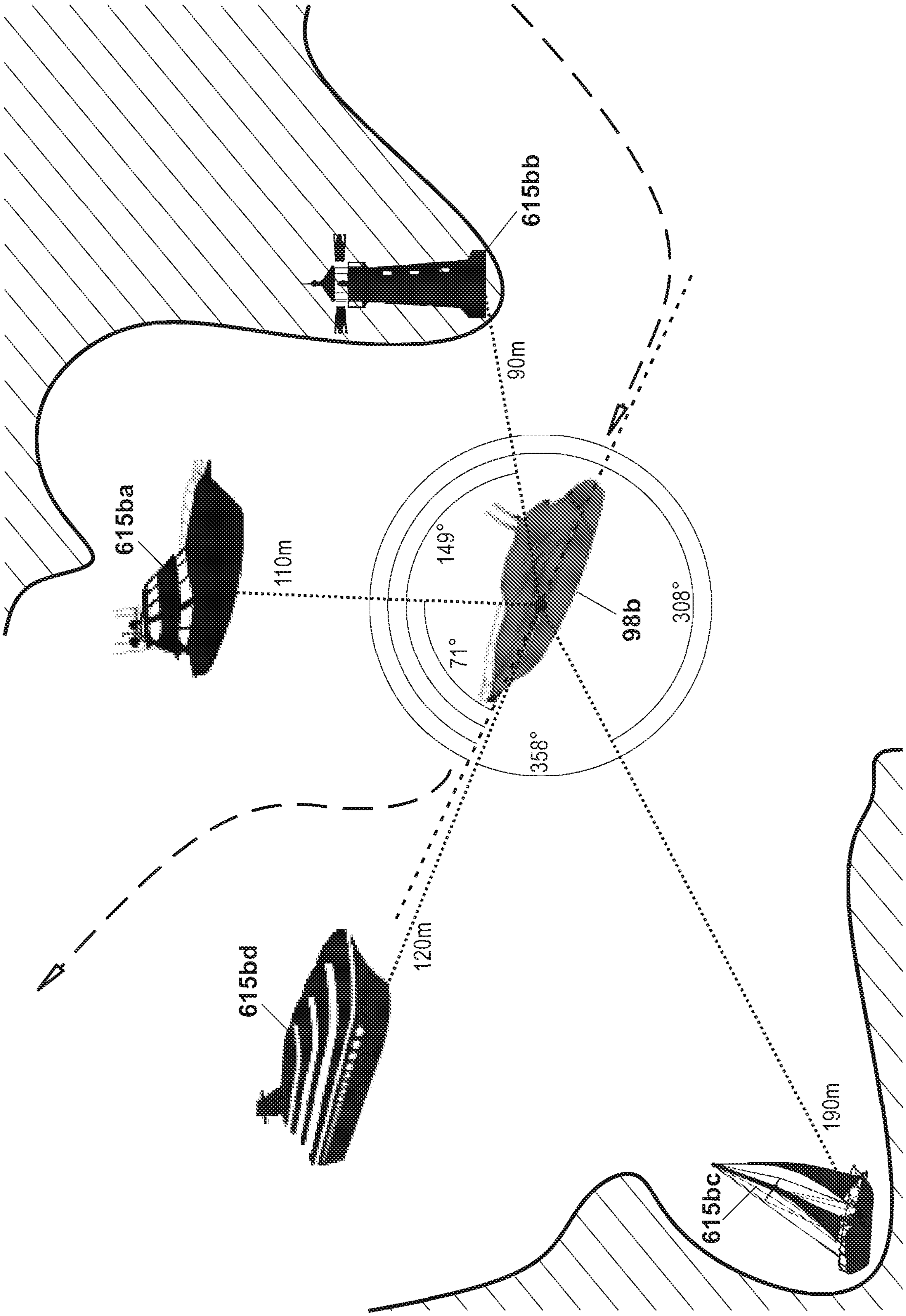
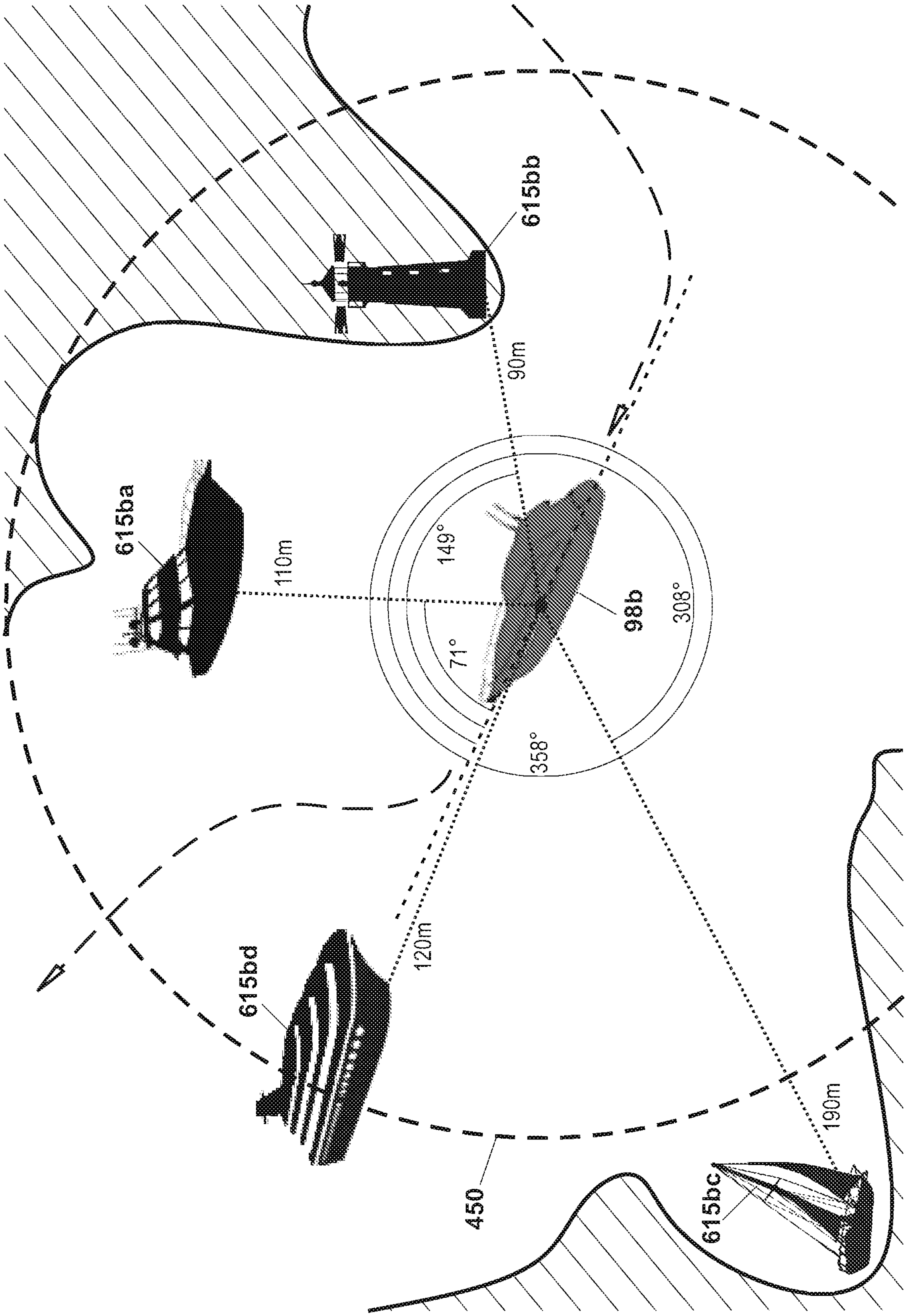


FIG. 39



615bb
90m
615ba
110m
149°
71°
98b
308°
358°
120m
615bd
190m
450
615bc

FIG. 40

Electronic Patent Application Fee Transmittal

Application Number:	
Filing Date:	
Title of Invention:	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION
First Named Inventor/Applicant Name:	Jasmin Cosic
Filer:	Jasmin Cosic
Attorney Docket Number:	

Filed as Small Entity

Filing Fees for **Utility under 35 USC 111(a)**

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Basic Filing:				
UTILITY FILING FEE (ELECTRONIC FILING)	4011	1	70	70
UTILITY SEARCH FEE	2111	1	300	300
UTILITY EXAMINATION FEE	2311	1	360	360
Pages:				
UTILITY APPL SIZE FEE PER 50 SHEETS >100	2081	1	200	200
Claims:				
Miscellaneous-Filing:				
Petition:				

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Patent-Appeals-and-Interference:				
Post-Allowance-and-Post-Issuance:				
Extension-of-Time:				
Miscellaneous:				
		Total in USD ($)		**930**

Electronic Acknowledgement Receipt

EFS ID:	27391591
Application Number:	15340991
International Application Number:	
Confirmation Number:	1993
Title of Invention:	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION
First Named Inventor/Applicant Name:	Jasmin Cosic
Customer Number:	116094
Filer:	Jasmin Cosic
Filer Authorized By:	
Attorney Docket Number:	
Receipt Date:	02-NOV-2016
Filing Date:	
Time Stamp:	02:17:15
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	yes
Payment Type	CARD
Payment was successfully received in RAM	$930
RAM confirmation Number	110216INTEFSW02222800
Deposit Account	
Authorized User	

The Director of the USPTO is hereby authorized to charge indicated fees and credit any overpayment as follows:

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1	Abstract	ABSTRACT_flattened.pdf	220691 da3ae9f7d0b2f5ba2cf0ec8b463a23fc7d46 5c06	no	1
Warnings:					
Information:					
2	Application Data Sheet	ADS.pdf	1822777 9f1905405423b472d1600b2612af4322cdb b8438	no	8
Warnings:					
Information:					
3	Oath or Declaration filed	DECLARATION_flattened.pdf	80488 20389855346f595c777e59543f418bd69a3 e225b	no	1
Warnings:					
Information:					
4	Specification	SPECIFICATION_flatenned.pdf	11824929 7189a9698cb43429c7efec2330d9b2154bc 58743	no	152
Warnings:					
Information:					
5	Claims	CLAIMS_flattened.pdf	2974744 2fd110cc8195f4644b63a7b3c8a86585bcd6 51e5	no	8
Warnings:					
Information:					
6	Drawings-only black and white line drawings	DRAWINGS_flattened.pdf	1137891 0551a2e3e7754760ebd68a3a1041bf9c4a8 edd7e	no	40
Warnings:					
The page size in the PDF is too large. The pages should be 8.5 x 11 or A4. If this PDF is submitted, the pages will be resized upon entry into the Image File Wrapper and may affect subsequent processing					

Information:					
7	Fee Worksheet (SB06)	fee-info.pdf	36505 c5001ebf77638214a3b771f56d2dfb907a865a87	no	2
Warnings:					
Information:					
		Total Files Size (in bytes):	18098025		

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.

National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

DocCode - SCORE

SCORE Placeholder Sheet for IFW Content

Application Number: 15340991 Document Date: 11/02/2016

The presence of this form in the IFW record indicates that the following document type was received in electronic format on the date identified above. This content is stored in the SCORE database.

- Drawings – Other than Black and White Line Drawings

Since this was an electronic submission, there is no physical artifact folder, no artifact folder is recorded in PALM, and no paper documents or physical media exist. The TIFF images in the IFW record were created from the original documents that are stored in SCORE.

To access the documents in the SCORE database, refer to instructions below.

At the time of document entry (noted above):

- Examiners may access SCORE content via the eDAN interface.
- Other USPTO employees can bookmark the current SCORE URL (http://Score.uspto.gov/ScoreAccessWeb/).
- External customers may access SCORE content via the Public and Private PAIR interfaces.

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS

Remove

Examiner Initial*	Cite No	Patent Number	Kind Code[1]	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	4370707		1983-01-25	Phillips , et al.	
	2	4730315		1988-03-08	Saito , et al.	
	3	4860203		1989-08-22	Corrigan , et al.	
	4	5602982		1997-02-11	Judd , et al.	
	5	6026234		2000-02-15	Hanson , et al.	
	6	6088731		2000-07-11	Kiraly , et al.	
	7	6106299		2000-08-22	Ackermann , et al.	
	8	6126330		2000-10-03	Knight	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	9	6314558		2001-11-06	Angel , et al.	
	10	6643842		2003-11-04	Angel , et al.	
	11	6728689		2004-04-27	Drissi , et al.	
	12	6735632		2004-05-11	Kiraly , et al.	
	13	6801912		2004-10-05	Moskowitz , et al.	
	14	6915105		2005-07-05	Masuda	
	15	6973446		2005-12-06	Mamitsuka , et al.	
	16	7017153		2006-03-21	Gouriou , et al.	
	17	7052277		2006-05-30	Kellman	
	18	7222127		2007-05-22	Bem , et al.	
	19	7240335		2007-07-03	Angel , et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	20	7249349		2007-07-24	Hundt , et al.	
	21	7424705		2008-09-09	Lewis , et al.	
	22	7478371		2009-01-13	Gove	
	23	7484205		2009-01-27	Venkatapathy	
	24	7721218		2010-05-18	Awe , et al.	
	25	7765537		2010-07-27	Havin , et al.	
	26	7797259		2010-09-14	Jiang , et al.	
	27	7840060		2010-11-23	Podilchuk	
	28	7925984		2011-04-12	Awe , et al.	
	29	7987144		2011-07-26	Drissi , et al.	
	30	8005828		2011-08-23	Buchner , et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	31	8019699		2011-09-13	Baxter	
	32	8078556		2011-12-13	Adi , et al.	
	33	8090669		2012-01-03	Shahani , et al.	
	34	8137112		2012-03-20	Woolf , et al.	
	35	8166463		2012-04-24	Gill , et al.	
	36	8195674		2012-06-05	Bem , et al.	
	37	8196119		2012-06-05	Gill , et al.	
	38	8244730		2012-08-14	Gupta	
	39	8251704		2012-08-28	Woolf , et al.	
	40	8266608		2012-09-11	Hecht , et al.	
	41	8364612		2013-01-29	Van Gael , et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	42	8386401		2013-02-26	Virkar , et al.	
	43	8397227		2013-03-12	Fan , et al.	
	44	8464225		2013-06-11	Greifeneder	
	45	8549359		2013-10-01	Zheng	
	46	8589414		2013-11-19	Waite , et al.	
	47	8655260		2014-02-18	Yueh , et al.	
	48	8667472		2014-03-04	Molinari	

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS Remove

Examiner Initial*	Cite No	Publication Number	Kind Code[1]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	20120167057		2012-06-28	Schmich; Christopher P. ; et al.	
	2	20120150773		2012-06-14	DiCorpo; Phillip ; et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	3	20080254429		2008-10-16	Woolf; Susan D. ; et al.	
	4	20040117771		2004-06-17	Venkatapathy, Ramanathan	
	5	20130238533		2013-09-12	VIRKAR; Hemant ; et al.	
	6	20060190930		2006-08-24	Hecht; Daniel M. ; et al.	
	7	20070006159		2007-01-04	Hecht; Daniel Michael ; et al.	
	8	20080281764		2008-11-13	Baxter; Jonathan	
	9	20100138370		2010-06-03	Wu; Wang ; et al.	
	10	20110218672		2011-09-08	Maisonnier; Bruno ; et al.	

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number[3]	Country Code[2] i	Kind Code[4]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T[5]
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T5
	1	About Event Tracing, retrieved from <URL: http://msdn.microsoft.com/en-us/library/aa363668(d=default,l=en-us, v=vs.85).aspx> on Jan 12, 2014, 2 pages	
	2	Add dynamic Java code to your application, retrieved from <URL: http://www.javaworld.com/article/2071777/design-patterns/add-dynamic-java-code-to-your-application.html> on May 13, 2014, 6 pages	
	3	BCEL, retrieved from <URL: http://commons.apache.org> on May 13, 2014, 2 pages	
	4	Artificial intelligence, retrieved from <URL: http://wikipedia.com> on May 21, 2014, 28 pages	
	5	Artificial neural network, retrieved from <URL: http://wikipedia.com> on May 21, 2014, 12 pages	
	6	Branch (computer science), retrieved from <URL: http://wikipedia.com> on May 6, 2014, 2 pages	
	7	Branch trace, retrieved from <URL: http://wikipedia.com> on May 28, 2014, 2 pages	
	8	C Function Call Conventions and the Stack, retrieved from <URL: http://www.csee.umbc.edu/~chang/cs313.s02/stack.shtml> on Jan 11, 2014, 4 pages	
	9	Call stack, retrieved from <URL: http://wikipedia.com> on Jan 11, 2014, 5 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	10	Program compilation and execution flow, retrieved from <URL: http://cs.stackexchange.com/questions/6187/program-compilation-and-execution-flow> on Jan 9, 2014, 2 pages	
	11	Control unit, retrieved from <URL: http://wikipedia.com> on May 25, 2014, 2 pages	
	12	Create dynamic applications with javax.tools, retrieved from <URL: http://www.ibm.com/developerworks/library/j-jcomp/> on May 13, 2014, 11 pages	
	13	Creating a Debugging and Profiling Agent with JVMTI, retrieved from <URL: http://www.oracle.com/technetwork/articles/java/jvmti-136367.html> on Jan 11, 2014, 18 pages	
	14	Decision tree learning, retrieved from <URL: http://wikipedia.com> on Jun 24, 2014, 5 pages	
	15	Dynamic Bytecode Instrumentation, retrieved from <URL: http://www.drdobbs.com/tools/dynamic-bytecode-instrumentation/184406433> on Apr 26, 2014, 6 pages	
	16	Dynamic programming language, retrieved from <URL: http://wikipedia.com> on May 7, 2014, 4 pages	
	17	Dynamic recompilation, retrieved from <URL: http://wikipedia.com> on May 19, 2014, 4 pages	
	18	eval(), retrieved from <URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/eval> on May 7, 2014, 13 pages	
	19	Function, retrieved from <URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Function> on May 7, 2014, 11 pages	
	20	FunctionEnter2 Function, retrieved from <URL: https://msdn.microsoft.com/en-us/library/aa964981(d=default,l=en-us,v=vs.110).aspx> on Mar 19, 2014, 2 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	21	FunctionLeave2 Function, retrieved from <URL: https://msdn.microsoft.com/en-us/library/aa964942(d=default,l=en-us,v=vs.110).aspx> on Mar 19, 2014, 1 pages	
	22	Heuristic, retrieved from <URL: http://wikipedia.com> on May 22, 2014, 5 pages	
	23	How to: Configure Trace Switches, retrieved from <URL: https://msdn.microsoft.com/en-us/library/vstudio/t06xyy08(d=default,l=en-us,v=vs.100).aspx> on Mar 19, 2014, 3 pages	
	24	Indirect branch, retrieved from <URL: http://wikipedia.com> on May 5, 2014, 1 pages	
	25	Instruction cycle, retrieved from <URL: http://wikipedia.com> on Apr 30, 2014, 2 pages	
	26	Instruction set, retrieved from <URL: http://wikipedia.com> on Apr 29, 2014, 6 pages	
	27	Interrupt, retrieved from <URL: http://wikipedia.com> on May 15, 2014, 5 pages	
	28	Introduction to Instrumentation and Tracing, retrieved from <URL: http://msdn.microsoft.com/en-us/library/aa983649(d=default,l=en-us,v=vs.71).aspx> on Jan 10, 2014, 2 pages	
	29	Java bytecode, retrieved from <URL: http://wikipedia.com> on May 27, 2014, 4 pages	
	30	Package java.util.logging, retrieved from <URL: http://docs.oracle.com/javase/1.5.0/docs/api/java/util/logging/package-summary.html> on Mar 19, 2014, 2 pages	
	31	javac - Java programming language compiler, retrieved from <URL: http://docs.oracle.com/javase/7/docs/technotes/tools/windows/javac.html> on May 13, 2014, 12 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	32	Interface JavaCompiler, retrieved from <URL: http://wikipedia.com> on May 13, 2014, 4 pages	
	33	Javassist, retrieved from <URL: http://wikipedia.com> on May 13, 2014, 2 pages	
	34	Just-in-time compilation, retrieved from <URL: http://wikipedia.com> on May 19, 2014, 5 pages	
	35	JVM Tool Interface, retrieved from <URL: http://docs.oracle.com/javase/7/docs/platform/jvmti/jvmti.html> on Apr 26, 2014, 127 pages	
	36	Machine code, retrieved from <URL: http://wikipedia.com> on Apr 30, 2014, 3 pages	
	37	Metaprogramming, retrieved from <URL: http://wikipedia.com> on May 8, 2014, 3 pages	
	38	Microsoft.VisualBasic.Logging Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/ms128122(d=default,l=en-us,v=vs.110).aspx> on Jan 13, 2014, 1 pages	
	39	Pin (computer program), retrieved from <URL: http://wikipedia.com> on May 6, 2014, 3 pages	
	40	Processor register, retrieved from <URL: http://wikipedia.com> on Apr 29, 2014, 4 pages	
	41	Profiling (computer programming), retrieved from <URL: http://wikipedia.com> on Jan 10, 2014, 4 pages	
	42	Profiling Overview, retrieved from <URL: http://msdn.microsoft.com/en-us/library/bb384493(d=default,l=en-us,v=vs.110).aspx> on Jan 12, 2014, 6 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	43	Program counter, retrieved from <URL: http://wikipedia.com> on Apr 24, 2014, 2 pages	
	44	ptrace, retrieved from <URL: http://wikipedia.com> on Apr 24, 2014, 2 pages	
	45	Reflection (computer programming), retrieved from <URL: http://wikipedia.com> on May 7, 2014, 5 pages	
	46	Class Runtime, retrieved from <URL: http://docs.oracle.com/javase/7/docs/api/java/lang/Runtime.html> on Jan 10, 2014, 10 pages	
	47	Self-modifying code, retrieved from <URL: http://wikipedia.com> on Jan 12, 2014, 6 pages	
	48	Subjective logic, retrieved from <URL: http://wikipedia.com> on May 22, 2014, 5 pages	
	49	System.CodeDom.Compiler Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/z6b99ydt(d=default,l=en-us,v=vs.110).aspx> on May 13, 2014, 2 pages	
	50	System.Reflection.Emit Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/xd5fw18y(d=default,l=en-us,v=vs.110).aspx> on May 13, 2014, 2 pages	

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature		Date Considered	

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Jasmin Cosic/	Date (YYYY-MM-DD)	2016-11-10
Name/Print	Jasmin Cosic	Registration Number	

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS

Remove

Examiner Initial*	Cite No	Patent Number	Kind Code[1]	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1					

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS

Remove

Examiner Initial*	Cite No	Publication Number	Kind Code[1]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1					

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS

Remove

Examiner Initial*	Cite No	Foreign Document Number[3]	Country Code[2] i	Kind Code[4]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T[5]
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS

Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T[5]

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	1	Tracing (software), retrieved from <URL: http://wikipedia.com> on Jan 10, 2014, 3 pages	
	2	Tree (data structure), retrieved from <URL: http://wikipedia.com> on Jun 24, 2014, 6 pages	
	3	PTRACE(2), retrieved from <URL: http://unixhelp.ed.ac.uk/CGI/man-cgi?ptrace> on Mar 19, 2014, 5 pages	
	4	Wevtutil, retrieved from <URL: http://technet.microsoft.com/en-us/library/cc732848(d=default,l=en-us,v=ws.11).aspx> on Apr 28, 2014, 5 pages	
	5	Intel Processor Trace, retrieved from <URL: https://software.intel.com/en-us/blogs/2013/09/18/processor-tracing> on Apr 28, 2014, 3 pages	
	6	YOUNGHOON JUNG, JAVA DYNAMICS Reflection and a lot more, Oct 10, 2012, 55 pages, Columbia University	
	7	AMITABH SRIVASTAVA, ALAN EUSTACE, ATOM A System for Building Customized Program Analysis Tools, May 3, 2004, 12 pages	
	8	MATHEW SMITHSON, KAPIL ANAND, APARNA KOTHA, KHALED ELWAZEER, NATHAN GILES, RAJEEV BARUA, Binary Rewriting without Relocation Information, Nov 10, 2010, 11 pages, University of Maryland	
	9	MAREK OLSZEWSKI, KEIR MIERTE, ADAM CZAJKOWSKI, ANGELA DEMLE BROWN, JIT Instrumentation - A Novel Approach To Dynamically Instrument Operating Systems, Feb 12, 2007, 14 pages, University of Toronto	

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature		Date Considered	

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Jasmin Cosic/	Date (YYYY-MM-DD)	2016-11-10
Name/Print	Jasmin Cosic	Registration Number	

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS Remove

Examiner Initial*	Cite No	Patent Number	Kind Code[1]	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1					

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS Remove

Examiner Initial*	Cite No	Publication Number	Kind Code[1]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	20030026588		2003-02-06	Elder, James H. ; et al.	
	2	20040249774		2004-12-09	Caid, William R. ; et al.	
	3	20040267521		2004-12-30	Cutler, Ross ; et al.	
	4	20050245303		2005-11-03	Graepel, Kurt Hartwig ; et al.	
	5	20060047612		2006-03-02	Stanley; Kenneth O. ; et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	6	20070058856		2007-03-15	Boregowda; Lokesh R. ; et al.	
	7	20070106633		2007-05-10	Reiner; Bruce	
	8	20080144893		2008-06-19	GUO; Chun Biao ; et al.	
	9	20080288259		2008-11-20	Chambers; Robert L. ; et al.	
	10	20090110061		2009-04-30	LEE; Kang Yong ; et al.	
	11	20090131152		2009-05-21	Busse; Martin A.	
	12	20090141969		2009-06-04	Yu; Kai ; et al.	
	13	20090222388		2009-09-30	Hua; Wei ; et al.	
	14	20090324010		2009-12-31	Hou; Billy	
	15	20100033780		2010-02-11	Gitter; Kurt	
	16	20100063949		2010-03-11	EATON; JOHN ERIC ; et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	17	20100241595		2010-09-23	Felsher; David Paul	
	18	20100278420		2010-11-04	Shet; Vinay Damodar ; et al.	
	19	20110030031		2011-02-03	Lussier; Paul ; et al.	
	20	20130156345		2013-06-20	Shmunk; Dmitry Valerievich	
	21	20130159021		2013-06-20	Felsher; David Paul	
	22	20130278631		2013-10-24	Border; John N. ; et al.	
	23	20140052717		2014-02-20	FRANKS; Kasian ; et al.	
	24	20140161250		2014-06-12	Leeds; Richard ; et al.	
	25	20140177946		2014-06-26	LIM; Kil-Taek ; et al.	
	26	20140207580		2014-06-24	Minnis; William Charles ; et al.	
	27	20150039304		2015-02-05	Wein; Ron	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	28	20150055821		2015-02-26	Fotland; David Allen	

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number3	Country Code2 i	Kind Code4	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T5
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T5
	1	ABOUT | OpenCV, retrieved from <URL: http://opencv.org/about.html> on Dec 13, 2014, 1 pages	
	2	Animetrics, Inc. | 3D Facial Recognition, retrieved from <URL: http://animetrics.com/> on Dec 13, 2014, 2 pages	
	3	Bag-of-words model, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	4	Beier–Neely morphing algorithm, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 1 pages	
	5	Computer vision, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 7 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	
Application Number	15340991
Filing Date	2016-11-02
First Named Inventor	Jasmin Cosic
Art Unit	
Examiner Name	
Attorney Docket Number	

	6	Convolutional neural network, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 5 pages	
	7	Decimation (signal processing), retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 3 pages	
	8	Digital image processing, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 3 pages	
	9	Dissolve (filmmaking), retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 2 pages	
	10	Facial recognition system, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 6 pages	
	11	Feature detection (computer vision), retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 3 pages	
	12	Feature extraction, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	13	Gesture recognition, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 5 pages	
	14	GrabCut, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 1 pages	
	15	Image processing, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	16	Image segmentation, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 12 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15340991	
	Filing Date	2016-11-02	
	First Named Inventor	Jasmin Cosic	
	Art Unit		
	Examiner Name		
	Attorney Docket Number		

	17	Image warping, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 2 pages	
	18	Inbetweening, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 2 pages	
	19	Interpolation, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 5 pages	
	20	Language model, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 4 pages	
	21	List of speech recognition software, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 4 pages	
	22	Livewire Segmentation Technique, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 2 pages	
	23	Morphing, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 2 pages	
	24	Motion estimation, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	25	Muse: Face Morph || Mesh Warping, retrieved from <URL: http://alexwolfe.blogspot.com/2011/10/face-morph-mesh-warping.html> on Nov 19, 2015, 2 pages	
	26	n-gram, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 5 pages	
	27	Optical flow, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 4 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	28	Outline of object recognition, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 7 pages	
	29	Phonotactics, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 3 pages	
	30	Recurrent neural network, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 5 pages	
	31	Sample rate conversion, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 2 pages	
	32	Simple interactive object extraction, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	33	Speech recognition, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 12 pages	
	34	Speech segmentation, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 4 pages	
	35	Welcome to recognize-speech.com, retrieved from <URL: http://recognize-speech.com/> on Oct 18, 2015, 1 pages	
	36	Introduction Speech, retrieved from <URL: http://recognize-speech.com/speech> on Oct 18, 2015, 1 pages	
	37	Preprocessing, retrieved from <URL: http://recognize-speech.com/preprocessing> on Oct 18, 2015, 4 pages	
	38	Feature Extraction, retrieved from <URL: http://recognize-speech.com/feature-extraction> on Oct 18, 2015, 3 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	39	Acoustic model, retrieved from <URL: http://recognize-speech.com/acoustic-model> on Oct 18, 2015, 2 pages	
	40	Video content analysis, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	41	Video tracking, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 2 pages	
	42	Andrej Karpathy, Thomas Leung, George Toderici, Rahul Sukthankar, Sanketh Shetty, Li Fei-Fei, Large-scale Video Classification with Convolutional Neural Networks, Apr 14, 2014, 8 pages, Stanford University	
	43	Karen Simonyan, Andrew Zisserman, Two-Stream Convolutional Networks for Action Recognition in Videos, Nov 13, 2014, 11 pages, University of Oxford	

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature		Date Considered	

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Jasmin Cosic/	Date (YYYY-MM-DD)	2016-11-10
Name/Print	Jasmin Cosic	Registration Number	

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS

Remove

Examiner Initial*	Cite No	Patent Number	Kind Code[1]	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	7113946		2006-09-26	Cosic	
	2	7117225		2006-10-03	Cosic	
	3	8335805		2012-12-18	Cosic	
	4	8417740		2013-04-09	Cosic	
	5	8572035		2013-10-29	Cosic	
	6	8655900		2014-02-18	Cosic	
	7	9047324		2015-06-02	Cosic	
	8	9282309		2016-03-08	Cosic	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	9	9298749		2016-03-29	Cosic	
	10	9367806		2016-06-14	Cosic	
	11	9443192		2016-09-13	Cosic	

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS Remove

Examiner Initial*	Cite No	Publication Number	Kind Code[1]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	20030065662		2003-04-03	Cosic	
	2	20040194017		2004-09-30	Cosic	
	3	20050149517		2005-07-07	Cosic	
	4	20050149542		2005-07-07	Cosic	
	5	20050289105		2005-12-29	Cosic	
	6	20100023541		2010-01-28	Cosic	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	7	20100082536		2010-04-01	Cosic	
	8	20130218932		2013-08-22	Cosic	
	9	20130226974		2013-08-29	Cosic	
	10	20160140999		2016-05-19	Cosic	
	11	20160142650		2016-05-19	Cosic	
	12	20160246819		2016-08-25	Cosic	
	13	20160246850		2016-08-25	Cosic	
	14	20160246868		2016-08-25	Cosic	
	15	20160292185		2016-10-06	Cosic	

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number[3]	Country Code[2] i	Kind Code[4]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T[5]

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T5
	1		

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature		Date Considered	

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Jasmin Cosic/	Date (YYYY-MM-DD)	2016-11-10
Name/Print	Jasmin Cosic	Registration Number	

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Electronic Acknowledgement Receipt

EFS ID:	27480807
Application Number:	15340991
International Application Number:	
Confirmation Number:	1993
Title of Invention:	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION
First Named Inventor/Applicant Name:	Jasmin Cosic
Customer Number:	116094
Filer:	Jasmin Cosic
Filer Authorized By:	
Attorney Docket Number:	
Receipt Date:	10-NOV-2016
Filing Date:	
Time Stamp:	20:16:32
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	no

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1	Information Disclosure Statement (IDS) Form (SB08)	IDS_Part1.pdf	615743 e8ef6a4647288aeeb4669422b73e856fbee955d6	no	13

Warnings:

Information:

2	Information Disclosure Statement (IDS) Form (SB08)	IDS_Part2.pdf	613126 c897c545b8c95a31e6ac9a660e4f9462410b4212	no	5

Warnings:

Information:

A U.S. Patent Number Citation or a U.S. Publication Number Citation is required in the Information Disclosure Statement (IDS) form for autoloading of data into USPTO systems. You may remove the form to add the required data in order to correct the Informational Message if you are citing U.S. References. If you chose not to include U.S. References, the image of the form will be processed and be made available within the Image File Wrapper (IFW) system. However, no data will be extracted from this form. Any additional data such as Foreign Patent Documents or Non Patent Literature will be manually reviewed and keyed into USPTO systems.

3	Information Disclosure Statement (IDS) Form (SB08)	IDS_Part3.pdf	614630 049e1bd89f595efb755d3fb7bfe296fc8a989a2b	no	10

Warnings:

Information:

4	Information Disclosure Statement (IDS) Form (SB08)	IDS_Part4.pdf	612917 e10b0c0937c6bb310686e40b525d3c11a97969f0	no	6

Warnings:

Information:

5	Non Patent Literature	NPL_Part1.pdf	19779115 ee07479784f30836d39a22bda6b5ec735694f56e	no	380

Warnings:

The page size in the PDF is too large. The pages should be 8.5 x 11 or A4. If this PDF is submitted, the pages will be resized upon entry into the Image File Wrapper and may affect subsequent processing

Information:

6	Non Patent Literature	NPL_Part2.pdf	5879608 8fd00edc18b82bf74a8ebf692fa5e39b231f3d56	no	114

Warnings:

The page size in the PDF is too large. The pages should be 8.5 x 11 or A4. If this PDF is submitted, the pages will be resized upon entry into the Image File Wrapper and may affect subsequent processing

Information:

7	Non Patent Literature	NPL_Part3.pdf	16903333 327e784f4ab311324ac99b3c6dd8b06ae0b170c7	no	160

Warnings:

Information:

Total Files Size (in bytes):	45018472

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.

National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

PATENT APPLICATION FEE DETERMINATION RECORD

Substitute for Form PTO-875

Application or Docket Number
15/340,991

APPLICATION AS FILED - PART I

FOR	NUMBER FILED (Column 1)	NUMBER EXTRA (Column 2)	SMALL ENTITY RATE($)	SMALL ENTITY FEE($)	OR	OTHER THAN SMALL ENTITY RATE($)	OTHER THAN SMALL ENTITY FEE($)
BASIC FEE (37 CFR 1.16(a), (b), or (c))	N/A	N/A	N/A	70		N/A	
SEARCH FEE (37 CFR 1.16(k), (i), or (m))	N/A	N/A	N/A	300		N/A	
EXAMINATION FEE (37 CFR 1.16(o), (p), or (q))	N/A	N/A	N/A	360		N/A	
TOTAL CLAIMS (37 CFR 1.16(i))	20 minus 20 =	*	x 40 =	0.00	OR		
INDEPENDENT CLAIMS (37 CFR 1.16(h))	3 minus 3 =	*	x 210 =	0.00			
APPLICATION SIZE FEE (37 CFR 1.16(s))	If the specification and drawings exceed 100 sheets of paper, the application size fee due is $310 ($155 for small entity) for each additional 50 sheets or fraction thereof. See 35 U.S.C. 41(a)(1)(G) and 37 CFR 1.16(s).			200			
MULTIPLE DEPENDENT CLAIM PRESENT (37 CFR 1.16(j))				0.00			
			TOTAL	930		TOTAL	

* If the difference in column 1 is less than zero, enter "0" in column 2.

APPLICATION AS AMENDED - PART II

AMENDMENT A

	CLAIMS REMAINING AFTER AMENDMENT (Column 1)		HIGHEST NUMBER PREVIOUSLY PAID FOR (Column 2)	PRESENT EXTRA (Column 3)	SMALL ENTITY RATE($)	SMALL ENTITY ADDITIONAL FEE($)	OR	OTHER THAN SMALL ENTITY RATE($)	OTHER THAN SMALL ENTITY ADDITIONAL FEE($)
Total (37 CFR 1.16(i))	*	Minus	**	=	x =		OR	x =	
Independent (37 CFR 1.16(h))	*	Minus	***	=	x =		OR	x =	
Application Size Fee (37 CFR 1.16(s))									
FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))							OR		
					TOTAL ADD'L FEE		OR	TOTAL ADD'L FEE	

AMENDMENT B

	CLAIMS REMAINING AFTER AMENDMENT (Column 1)		HIGHEST NUMBER PREVIOUSLY PAID FOR (Column 2)	PRESENT EXTRA (Column 3)	SMALL ENTITY RATE($)	SMALL ENTITY ADDITIONAL FEE($)	OR	OTHER THAN SMALL ENTITY RATE($)	OTHER THAN SMALL ENTITY ADDITIONAL FEE($)
Total (37 CFR 1.16(i))	*	Minus	**	=	x =		OR	x =	
Independent (37 CFR 1.16(h))	*	Minus	***	=	x =		OR	x =	
Application Size Fee (37 CFR 1.16(s))									
FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))							OR		
					TOTAL ADD'L FEE		OR	TOTAL ADD'L FEE	

* If the entry in column 1 is less than the entry in column 2, write "0" in column 3.
** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, enter "20".
*** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, enter "3".
The "Highest Number Previously Paid For" (Total or Independent) is the highest found in the appropriate box in column 1.

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NUMBER	FILING or 371(c) DATE	GRP ART UNIT	FIL FEE REC'D	ATTY.DOCKET.NO	TOT CLAIMS	IND CLAIMS
15/340,991	11/02/2016		930		20	3

CONFIRMATION NO. 1993

FILING RECEIPT

OC000000087312368

116094
Jasmin Cosic
108 Woodbury Street
Pawtucket, RI 02861

Date Mailed: 11/21/2016

Receipt is acknowledged of this non-provisional patent application. The application will be taken up for examination in due course. Applicant will be notified as to the results of the examination. Any correspondence concerning the application must include the following identification information: the U.S. APPLICATION NUMBER, FILING DATE, NAME OF APPLICANT, and TITLE OF INVENTION. Fees transmitted by check or draft are subject to collection. Please verify the accuracy of the data presented on this receipt. **If an error is noted on this Filing Receipt, please submit a written request for a Filing Receipt Correction. Please provide a copy of this Filing Receipt with the changes noted thereon. If you received a "Notice to File Missing Parts" for this application, please submit any corrections to this Filing Receipt with your reply to the Notice. When the USPTO processes the reply to the Notice, the USPTO will generate another Filing Receipt incorporating the requested corrections**

Inventor(s)

Jasmin Cosic, Miami, FL;

Applicant(s)

Jasmin Cosic, Miami, FL;

Power of Attorney: None

Domestic Applications for which benefit is claimed - None.
A proper domestic benefit claim must be provided in an Application Data Sheet in order to constitute a claim for domestic benefit. See 37 CFR 1.76 and 1.78.

Foreign Applications for which priority is claimed (You may be eligible to benefit from the **Patent Prosecution Highway** program at the USPTO. Please see http://www.uspto.gov for more information.) - None.
Foreign application information must be provided in an Application Data Sheet in order to constitute a claim to foreign priority. See 37 CFR 1.55 and 1.76.

Permission to Access Application via Priority Document Exchange: No

Permission to Access Search Results: No

Applicant may provide or rescind an authorization for access using Form PTO/SB/39 or Form PTO/SB/69 as appropriate.

If Required, Foreign Filing License Granted: 11/17/2016

The country code and number of your priority application, to be used for filing abroad under the Paris Convention, is **US 15/340,991**

Projected Publication Date: Request for Non-Publication Acknowledged

Non-Publication Request: Yes

Early Publication Request: No
**** SMALL ENTITY ****
Title

ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION

Preliminary Class

Statement under 37 CFR 1.55 or 1.78 for AIA (First Inventor to File) Transition Applications: No

PROTECTING YOUR INVENTION OUTSIDE THE UNITED STATES

Since the rights granted by a U.S. patent extend only throughout the territory of the United States and have no effect in a foreign country, an inventor who wishes patent protection in another country must apply for a patent in a specific country or in regional patent offices. Applicants may wish to consider the filing of an international application under the Patent Cooperation Treaty (PCT). An international (PCT) application generally has the same effect as a regular national patent application in each PCT-member country. The PCT process **simplifies** the filing of patent applications on the same invention in member countries, but **does not result** in a grant of "an international patent" and does not eliminate the need of applicants to file additional documents and fees in countries where patent protection is desired.

Almost every country has its own patent law, and a person desiring a patent in a particular country must make an application for patent in that country in accordance with its particular laws. Since the laws of many countries differ in various respects from the patent law of the United States, applicants are advised to seek guidance from specific foreign countries to ensure that patent rights are not lost prematurely.

Applicants also are advised that in the case of inventions made in the United States, the Director of the USPTO must issue a license before applicants can apply for a patent in a foreign country. The filing of a U.S. patent application serves as a request for a foreign filing license. The application's filing receipt contains further information and guidance as to the status of applicant's license for foreign filing.

Applicants may wish to consult the USPTO booklet, "General Information Concerning Patents" (specifically, the section entitled "Treaties and Foreign Patents") for more information on timeframes and deadlines for filing foreign patent applications. The guide is available either by contacting the USPTO Contact Center at 800-786-9199, or it can be viewed on the USPTO website at http://www.uspto.gov/web/offices/pac/doc/general/index.html.

For information on preventing theft of your intellectual property (patents, trademarks and copyrights), you may wish to consult the U.S. Government website, http://www.stopfakes.gov. Part of a Department of Commerce initiative, this website includes self-help "toolkits" giving innovators guidance on how to protect intellectual property in specific countries such as China, Korea and Mexico. For questions regarding patent enforcement issues, applicants may call the U.S. Government hotline at 1-866-999-HALT (1-866-999-4258).

LICENSE FOR FOREIGN FILING UNDER

Title 35, United States Code, Section 184

Title 37, Code of Federal Regulations, 5.11 & 5.15

GRANTED

The applicant has been granted a license under 35 U.S.C. 184, if the phrase "IF REQUIRED, FOREIGN FILING LICENSE GRANTED" followed by a date appears on this form. Such licenses are issued in all applications where the conditions for issuance of a license have been met, regardless of whether or not a license may be required as set forth in 37 CFR 5.15. The scope and limitations of this license are set forth in 37 CFR 5.15(a) unless an earlier license has been issued under 37 CFR 5.15(b). The license is subject to revocation upon written notification. The date indicated is the effective date of the license, unless an earlier license of similar scope has been granted under 37 CFR 5.13 or 5.14.

This license is to be retained by the licensee and may be used at any time on or after the effective date thereof unless it is revoked. This license is automatically transferred to any related applications(s) filed under 37 CFR 1.53(d). This license is not retroactive.

The grant of a license does not in any way lessen the responsibility of a licensee for the security of the subject matter as imposed by any Government contract or the provisions of existing laws relating to espionage and the national security or the export of technical data. Licensees should apprise themselves of current regulations especially with respect to certain countries, of other agencies, particularly the Office of Defense Trade Controls, Department of State (with respect to Arms, Munitions and Implements of War (22 CFR 121-128)); the Bureau of Industry and Security, Department of Commerce (15 CFR parts 730-774); the Office of Foreign AssetsControl, Department of Treasury (31 CFR Parts 500+) and the Department of Energy.

NOT GRANTED

No license under 35 U.S.C. 184 has been granted at this time, if the phrase "IF REQUIRED, FOREIGN FILING LICENSE GRANTED" DOES NOT appear on this form. Applicant may still petition for a license under 37 CFR 5.12, if a license is desired before the expiration of 6 months from the filing date of the application. If 6 months has lapsed from the filing date of this application and the licensee has not received any indication of a secrecy order under 35 U.S.C. 181, the licensee may foreign file the application pursuant to 37 CFR 5.15(b).

SelectUSA

The United States represents the largest, most dynamic marketplace in the world and is an unparalleled location for business investment, innovation, and commercialization of new technologies. The U.S. offers tremendous resources and advantages for those who invest and manufacture goods here. Through SelectUSA, our nation works to promote and facilitate business investment. SelectUSA provides information assistance to the international investor community; serves as an ombudsman for existing and potential investors; advocates on behalf of U.S. cities, states, and regions competing for global investment; and counsels U.S. economic development organizations on investment attraction best practices. To learn more about why the United States is the best country in the world to develop technology, manufacture products, deliver services, and grow your business, visit http://www.SelectUSA.gov or call +1-202-482-6800.

PATENT APPLICATION FEE DETERMINATION RECORD Substitute for Form PTO-875	Application or Docket Number 15/340,991	Filing Date 11/02/2016	☐ To be Mailed

ENTITY: ☐ LARGE ☑ SMALL ☐ MICRO

APPLICATION AS FILED - PART I

FOR	(Column 1) NUMBER FILED	(Column 2) NUMBER EXTRA	RATE ($)	FEE ($)
☐ BASIC FEE (37 CFR 1.16(a), (b), or (c))	N/A	N/A	N/A	
☐ SEARCH FEE (37 CFR 1.16(k), (i), or (m))	N/A	N/A	N/A	
☐ EXAMINATION FEE (37 CFR 1.16(o), (p), or (q))	N/A	N/A	N/A	
TOTAL CLAIMS (37 CFR 1.16(i))	minus 20 =	*	x $40 =	
INDEPENDENT CLAIMS (37 CFR 1.16(h))	minus 3 =	*	x $210 =	
☐ APPLICATION SIZE FEE (37 CFR 1.16(s))	If the specification and drawings exceed 100 sheets of paper, the application size fee due is $310 ($155 for small entity) for each additional 50 sheets or fraction thereof. See 35 U.S.C. 41(a)(1)(G) and 37 CFR 1.16(s).			
☐ MULTIPLE DEPENDENT CLAIM PRESENT (37 CFR 1.16(j))				
* If the difference in column 1 is less than zero, enter "0" in column 2.			TOTAL	

APPLICATION AS AMENDED - PART II

AMENDMENT	05/04/2019	(Column 1) CLAIMS REMAINING AFTER AMENDMENT		(Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR	(Column 3) PRESENT EXTRA	RATE ($)	ADDITIONAL FEE ($)
	Total (37 CFR 1.16(i))	* 20	Minus	** 20	= 0	x $50 =	0
	Independent (37 CFR 1.16(h))	* 3	Minus	*** 3	= 0	x $230 =	0
	☐ Application Size Fee (37 CFR 1.16(s))						
	☐ FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))						
						TOTAL ADD'L FEE	0

AMENDMENT		(Column 1) CLAIMS REMAINING AFTER AMENDMENT		(Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR	(Column 3) PRESENT EXTRA	RATE ($)	ADDITIONAL FEE ($)
	Total (37 CFR 1.16(i))	*	Minus	**	=	x $0 =	
	Independent (37 CFR 1.16(h))	*	Minus	***	=	x $0 =	
	☐ Application Size Fee (37 CFR 1.16(s))						
	☐ FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))						
						TOTAL ADD'L FEE	

* If the entry in column 1 is less than the entry in column 2, write "0" in column 3.	LIE
** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, enter "20".	/WILLIAM N PHILLIPS/
*** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, enter "3".	
The "Highest Number Previously Paid For" (Total or Independent) is the highest number found in the appropriate box in column 1.	

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Inventor: Jasmin Cosic Confirmation No.: 1993

Title: "ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION"

Serial No.: 15/340,991 Filed: November 2, 2016

Examiner: PARK, SOO JIN Group Art Unit: 2668

Via EFS-Web May 4, 2019

Mail Stop Amendment
COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, VA 22313-1450

PRELIMINARY AMENDMENT

Dear Commissioner:

Before examination on the merits, please amend the claims in the above-identified application as shown in the following listing of claims.

Electronic Acknowledgement Receipt

EFS ID:	35919394
Application Number:	15340991
International Application Number:	
Confirmation Number:	1993
Title of Invention:	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION
First Named Inventor/Applicant Name:	Jasmin Cosic
Customer Number:	116094
Filer:	Jasmin Cosic
Filer Authorized By:	
Attorney Docket Number:	
Receipt Date:	04-MAY-2019
Filing Date:	02-NOV-2016
Time Stamp:	22:35:52
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	no

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1		PRELIMINARY_AMENDMENT.pdf	40926 6bf4f7439b644297b38609aca6b1923e32fa2f87	yes	11

Multipart Description/PDF files in .zip description		
Document Description	**Start**	**End**
Applicant Arguments/Remarks Made in an Amendment	11	11
Claims	2	10
Preliminary Amendment	1	1

Warnings:

Information:

Total Files Size (in bytes):	40926

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.
National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.
New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

Inventor: Jasmin Cosic
Serial No.: 15/340,991
Filing Date: November 2, 2016

Remarks

In this preliminary amendment, the applicant cancels original claims 1-20 and presents for examination new claims 21-40. After entry of this preliminary amendment, claims 21-40 are pending. The undersigned may be contacted at (317) 772-1312 concerning this application.

I hereby certify that this correspondence is being submitted electronically via EFS Web to the United States Patent and Trademark Office.

By /Jasmin Cosic/
Jasmin Cosic

Date submitted: May 4, 2019

Respectfully submitted,

/Jasmin Cosic/

Jasmin Cosic

Listing of Claims

1 - 20 (Canceled)

21. (new) A system comprising:

one or more processor circuits;

a memory that stores at least a first correlation including a first one or more object representations correlated with a first one or more instruction sets for operating a first device, wherein the first device includes a first one or more sensors, and wherein the first one or more object representations represent a first one or more objects detected at least in part by the first one or more sensors; and

an artificial intelligence unit that:

generates or receives a second one or more object representations, wherein the second one or more object representations represent a second one or more objects;

determines the first one or more instruction sets for operating the first device based on at least partial match between the second one or more object representations and the first one or more object representations; and

in response to the determines of the artificial intelligence unit, causes the first device or a second device to perform one or more operations defined by the first one or more instruction sets for operating the first device at least by causing

the one or more processor circuits to execute the first one or more instruction sets for operating the first device.

22. (new) The system of claim 21, wherein at least one object of the first one or more objects and at least one object of the second one or more objects are the same.

23. (new) The system of claim 21, wherein the determines the first one or more instruction sets for operating the first device based on the at least partial match between the second one or more object representations and the first one or more object representations includes:

determining that a number of at least partially matching portions of the second one or more object representations and portions of the first one or more object representations exceeds a threshold number; or

determining that a percentage of at least partially matching portions of the second one or more object representations and portions of the first one or more object representations exceeds a threshold percentage.

24. (new) The system of claim 21, wherein the memory further stores at least a first knowledge cell, and wherein the first knowledge cell includes the first correlation.

25. (new) The system of claim 21, wherein the first one or more object representations include a first stream of one or more object representations, and wherein the second one or more object representations include a second stream of one or more object representations.

26. (new) The system of claim 21, wherein, to correlate the first one or more object representations with the first one or more instruction sets for operating the first device, a determination is made that the first one or more instruction sets for operating the first device temporally correspond to the first one or more object representations.

27. (new) The system of claim 21, wherein the artificial intelligence unit includes at least one selected from the group consisting of: a hardware element that is included in the one or more processor circuits, a hardware element that is included in another one or more processor circuits, a program operating on the one or more processor circuits, a program operating on another one or more processor circuits, and an element coupled to the one or more processor circuits, and wherein the first device includes: a robot, a vehicle, an appliance, an electronic device, or a mechanical machine, and wherein the second device includes: a robot, a vehicle, an appliance, an electronic device, or a mechanical machine.

28. (new) The system of claim 21, wherein the memory further stores at least a third correlation including a third one or more object representations correlated with a third one or more instruction sets for operating the first device, and wherein the third one or more object representations represent a third one or more objects detected at least in part by the first one or more sensors, and wherein a first connection is generated to connect the first correlation with the third correlation, and wherein the first correlation connected with the third correlation form at least a portion of a knowledge structure or a knowledgebase.

29. (new) The system of claim 21, wherein the memory further stores at least a third correlation including a third one or more object representations correlated with a third one or more instruction sets for operating the second device, and wherein the second device includes a second one or more sensors, and wherein the third one or more object representations represent a third one or more objects detected at least in part by the second one or more sensors.

30. (new) The system of claim 21, wherein at least a portion of the first correlation is learned in a learning process while the first device is at least partially operated by a user, and wherein the learning process includes:

generating or receiving the first one or more object representations; and

obtaining or receiving the first one or more instruction sets for operating the first device.

31. (new) The system of claim 21, wherein the artificial intelligence unit includes at least one selected from the group consisting of: at least a portion of an object processing unit, at least a portion of an acquisition interface, at least a portion of a modification interface, and at least a portion of a DCADO unit.

32. (new) The system of claim 21, wherein the second one or more objects include one or more objects detected at least in part by the first one or more sensors, and wherein the first one or more instruction sets for operating the first device are applied to the first device so that the first device performs the one or more operations defined by the first one or more instruction sets for operating the first device.

33. (new) The system of claim 21, wherein the second device includes a second one or more sensors, and wherein the second one or more objects include one or more objects detected at least in part by the second one or more sensors, and wherein the first one or more instruction sets for operating the first device are applied to the second device so that the second device performs the one or more operations defined by the first one or more instruction sets for operating the first device.

34. (new) The system of claim 21, wherein the second device includes a second one or more sensors, and wherein the second one or more objects include one or more objects detected at least in part by the second one or more sensors, and

wherein the first one or more instruction sets for operating the first device are modified and applied to the second device so that the second device performs the one or more operations defined by the modified first one or more instruction sets for operating the first device.

35. (new) A non-transitory machine readable medium having a program stored thereon that when executed by one or more processor circuits causes the one or more processor circuits to perform operations comprising:

accessing a memory that stores at least a first correlation including a first one or more object representations correlated with a first one or more instruction sets for operating a first device, wherein the first device includes a first one or more sensors, and wherein the first one or more object representations represent a first one or more objects detected at least in part by the first one or more sensors;

generating or receiving a second one or more object representations, wherein the second one or more object representations represent a second one or more objects;

determining the first one or more instruction sets for operating the first device based on at least partial match between the second one or more object representations and the first one or more object representations; and

in response to the determining, causing the first device or a second device to perform one or more operations defined by the first one or more instruction sets for operating the first device at least by causing the one or more processor

circuits or another one or more processor circuits to execute the first one or more instruction sets for operating the first device.

36. (new) The non-transitory machine readable medium of claim 35, wherein, to correlate the first one or more object representations with the first one or more instruction sets for operating the first device, a determination is made that the first one or more instruction sets for operating the first device temporally correspond to the first one or more object representations.

37. (new) The non-transitory machine readable medium of claim 35, wherein the second device includes a second one or more sensors, and wherein the second one or more objects include one or more objects detected at least in part by the second one or more sensors, and wherein the first one or more instruction sets for operating the first device are applied to the second device so that the second device performs the one or more operations defined by the first one or more instruction sets for operating the first device.

38. (new) A method comprising:

(a) accessing a memory that stores at least a first correlation including a first one or more object representations correlated with a first one or more instruction sets for operating a first device, wherein the first device includes a first one or more sensors, and wherein the first one or more object representations

represent a first one or more objects detected at least in part by the first one or more sensors, the accessing of (a) performed by one or more processor circuits;

(b) generating or receiving a second one or more object representations, wherein the second one or more object representations represent a second one or more objects, the generating or the receiving of (b) performed by the one or more processor circuits;

(c) determining the first one or more instruction sets for operating the first device based on at least partial match between the second one or more object representations and the first one or more object representations, the determining of (c) performed by the one or more processor circuits;

(d) executing the first one or more instruction sets for operating the first device, the executing of (d) performed by the one or more processor circuits or by another one or more processor circuits in response to the determining of (c); and

(e) performing, by the first device or by a second device, one or more operations defined by the first one or more instruction sets for operating the first device.

39. (new) The method of claim 38, wherein, to correlate the first one or more object representations with the first one or more instruction sets for operating the first device, a determination is made that the first one or more instruction sets for operating the first device temporally correspond to the first one or more object representations.

40. (new) The method of claim 38, wherein the second device includes a second one or more sensors, and wherein the second one or more objects include one or more objects detected at least in part by the second one or more sensors, and wherein the first one or more instruction sets for operating the first device are applied to the second device so that the second device performs the one or more operations defined by the first one or more instruction sets for operating the first device.

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS

Remove

Examiner Initial*	Cite No	Patent Number	Kind Code[1]	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	8996432		2015-03-31	Fu; Jicheng	
	2	6754631		2004-06-22	Din	
	3	9305216		2016-04-05	Mishra	
	4	5560011		1996-09-23	Uyama	
	5	6842877		2005-01-11	Robarts , et al.	
	6	7565340		2009-07-21	Herlocker , et al.	
	7	8261199		2012-09-04	Cradick , et al.	
	8	8266608		2012-09-11	Hecht , et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	9	9268454		2016-02-23	Hamilton, II , et al.	

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS

Remove

Examiner Initial*	Cite No	Publication Number	Kind Code1	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	20120290347		2012-11-15	ELAZOUNI; ASHRAF ; et al.	
	2	20150324685		2015-11-12	Bohn; Richard Esten ; et al.	
	3	20100114746		2010-05-06	Bobbitt; Russell Patrick ; et al.	
	4	20120284026		2012-11-08	Cardillo; Peter S. ; et al.	
	5	20140211988		2014-07-31	Fan; Quanfu ; et al.	
	6	20150006171		2015-01-01	Westby; Michael C. ; et al.	
	7	20150264306		2015-09-17	Marilly; Emmanuel ; et al.	
	8	20110085734		2011-04-14	Berg; Jared S. ; et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	9	20150269415		2015-09-24	Gelbman; Alexander	
	10	20160274187		2016-09-22	MENON; SANKARAN M. ; et al.	
	11	20150339213		2015-11-22	LEE; Christopher Stephen ; et al.	
	12	20150310041		2015-10-29	Kier; Scott ; et al.	
	13	20160328480		2016-11-10	Owens; Erich James ; et al.	
	14	20140075249		2014-03-13	SATO; Shuhei ; et al.	
	15	20110270794		2011-11-03	Drory; Tal ; et al.	
	16	20070050719		2007-03-01	Lui; Philip ; et al.	
	17	20070061735		2007-03-15	Hoffberg; Steven M. ; et al.	
	18	20060184410		2006-08-17	Ramamurthy; Shankar ; et al.	
	19	20060265406		2006-11-23	Chkodrov; Gueorgui B. ; et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	20	20090287643		2009-11-19	Corville; Allen O. ; et al.	
	21	20110007079		2011-01-13	Perez et al.	
	22	20070050606		2007-03-01	Ferren et al.	
	23	20090067727		2009-03-12	HIROHATA; Hitoshi	
	24	20130007532		2013-01-03	Miller et al.	
	25	20050240412		2005-10-27	Fujita	
	26	20160167226		2016-06-16	Schnittman	
	27	20090136095		2009-05-28	Marcon et al.	
	28	20080215508		2008-09-04	Hanneman; Jeffrey E. ; et al.	
	29	20090044113		2009-02-12	Jones; Scott T. ; et al.	
	30	20130278501		2013-10-24	BULZACKI; Adrian	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number3	Country Code2 i	Kind Code4	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T5
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T5
	1	Chen et al. "Case-Based Reasoning System and Artificial Neural Networks: A Review Neural Comput & Applic (2001) 10: pp 264-276, 13 pages	
	2	JOHN J. GREFENSTETTE, CONNIE LOGGIA RAMSEY, ALAN C. SCHULTZ, Learning Sequential Decision Rules Using Simulation Models and Competition, 1990, Navy Center for Applied Research in Artificial Intelligence, Naval Research Laboratory, Washington, DC, 27 pages	
	3	ALAN C. SCHULTZ, JOHN J. GREFENSTETTE, Using a Genetic Algorithm to Learn Behaviors for Autonomous Vehicles, 1992, Navy Center for Applied Research in Artificial Intelligence, Naval Research Laboratory, Washington, DC, 12 pages	
	4	Koppula et al., "Anticipating human activities using object affordances for reactive robotic response", IEEE TRAMI 2016, published 5 May 2015, 16 pages	
	5	Orme, "System design tips for entry level smartphones - part 3", found online at "https://community.arm.com/processors/b/blog/posts/system-design-tips-for-entry-level-smartphones---part-3", Oct 21, 2013, 12 pages	
	6	NOWOSTAWSKI, MARIUSZ et al., "Dynamic Demes Parallel Genetic Algorithm", May 13, 1999, 6 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	7	Stack Overflow, How can I quantify difference between two images?, accessed 2 August 2017 at https://stackoverflow.com/questions/189943/how-can-i-quantify-difference-between-two-images, 8 pages	
	8	SKOLICKI, ZBIGNIEW et al., "The Influence of Migration Sizes and Intervals on Island Models", June 29, 2005, 8 pages	
	9	Vahid Lari, et al., "Decentralized dynamic resource management support for massively parallel processor arrays", September 11, 2011, 8 pages	
	10	Vahid Lari, et al., "Distributed resource reservation in massively parallel processor arrays", May 16, 2011, 4 pages	
	11	Mohsen Hayati and Yazdan Shirvany, "Artificial Neural Network Approach for Short Term Load Forecasting for Illam Region", January 2007, 5 pages	
	12	Hasim Sak, Andrew Senior, and Francoise Beaufays, "Long Short-Term Memory Recurrent Neural Network Architectures for Large Scale Acoustic Modeling", January 2014, 5 pages	
	13	Jorg Walter and Klaus Schulten, "Implementation of self-organizing neural networks for visuo-motor control of an industrial robot", January 1993, 10 pages	
	14	Heikki Hyotyniemi and Aarno Lehtola, "A Universal Relation Database Interface for Knowledge Based Systems", April 1991, 5 pages	
	15	Mrissa, Michael, et al. "An avatar architecture for the web of things." IEEE Internet Computing 19.2 (2015): 30-38., 9 pages	
	16	Luck, Michael, and Ruth Aylett. "Applying artificial intelligence to virtual reality: Intelligent virtual environments." Applied Artificial Intelligence 14.1 (2000): 3-32., 30 pages	
	17	Terdjimi, Mehdi, et al. "An avatar-based adaptation workflow for the web of things." Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), 2016 IEEE 25th International Conference on. IEEE, 2016., 6 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	18	Bogdanovych, Anton, et al. "Authentic interactive reenactment of cultural heritage with 3D virtual worlds and artificial intelligence." Applied Artificial Intelligence 24.6 (2010): 617-647., 32 pages	
	19	Hernandez, Marco E. Perez, and Stephan Reiff-Marganiec. "Autonomous and self controlling smart objects for the future internet." Future internet of things and cloud (FiCloud), 2015 3rd international conference on. IEEE, 2015., 8 pages	
	20	Medini et al., "Building a Web of Things with Avatars", Managing the Web of Things (2017), 30 pages	

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature		Date Considered	

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Jasmin Cosic/	Date (YYYY-MM-DD)	2019-05-05
Name/Print	Jasmin Cosic	Registration Number	

This collection of information is required by 37 CFR 1.97 and 1.98. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 1 hour to complete, including gathering, preparing and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Electronic Acknowledgement Receipt

EFS ID:	35919853
Application Number:	15340991
International Application Number:	
Confirmation Number:	1993
Title of Invention:	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION
First Named Inventor/Applicant Name:	Jasmin Cosic
Customer Number:	116094
Filer:	Jasmin Cosic
Filer Authorized By:	
Attorney Docket Number:	
Receipt Date:	05-MAY-2019
Filing Date:	02-NOV-2016
Time Stamp:	15:18:25
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	no

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1	Information Disclosure Statement (IDS) Form (SB08)	IDS_Part5.pdf	619072 872308068d078ac226845930811367d9be024dd2	no	9

Warnings:

Information:

2	Non Patent Literature	NPL_Part5.pdf	25533925 58bed29843cae9904ec69a4efe47cc83b3ec e30a	no	254

Warnings:

The page size in the PDF is too large. The pages should be 8.5 x 11 or A4. If this PDF is submitted, the pages will be resized upon entry into the Image File Wrapper and may affect subsequent processing

Information:

Total Files Size (in bytes):	26152997

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.
National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.
New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

PTO/SB/06 (09-11)
Approved for use through 1/31/2014. OMB 0651-0032
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

PATENT APPLICATION FEE DETERMINATION RECORD Substitute for Form PTO-875	Application or Docket Number 15/340,991	Filing Date 11/02/2016	☐ To be Mailed

ENTITY: ☐ LARGE ☑ SMALL ☐ MICRO

APPLICATION AS FILED - PART I

FOR	(Column 1) NUMBER FILED	(Column 2) NUMBER EXTRA		RATE ($)	FEE ($)
☐ BASIC FEE (37 CFR 1.16(a), (b), or (c))	N/A	N/A		N/A	
☐ SEARCH FEE (37 CFR 1.16(k), (i), or (m))	N/A	N/A		N/A	
☐ EXAMINATION FEE (37 CFR 1.16(o), (p), or (q))	N/A	N/A		N/A	
TOTAL CLAIMS (37 CFR 1.16(i))	minus 20 =	*		x $40 =	
INDEPENDENT CLAIMS (37 CFR 1.16(h))	minus 3 =	*		x $210 =	
☐ APPLICATION SIZE FEE (37 CFR 1.16(s))	If the specification and drawings exceed 100 sheets of paper, the application size fee due is $310 ($155 for small entity) for each additional 50 sheets or fraction thereof. See 35 U.S.C. 41(a)(1)(G) and 37 CFR 1.16(s).				
☐ MULTIPLE DEPENDENT CLAIM PRESENT (37 CFR 1.16(j))					
* If the difference in column 1 is less than zero, enter "0" in column 2.				TOTAL	

APPLICATION AS AMENDED - PART II

AMENDMENT	05/27/2019	(Column 1) CLAIMS REMAINING AFTER AMENDMENT		(Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR	(Column 3) PRESENT EXTRA		RATE ($)	ADDITIONAL FEE ($)
	Total (37 CFR 1.16(i))	* 20	Minus	** 20	= 0		x $50 =	0
	Independent (37 CFR 1.16(h))	* 3	Minus	*** 3	= 0		x $230 =	0
	☐ Application Size Fee (37 CFR 1.16(s))							
	☐ FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))							
							TOTAL ADD'L FEE	0

AMENDMENT		(Column 1) CLAIMS REMAINING AFTER AMENDMENT		(Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR	(Column 3) PRESENT EXTRA		RATE ($)	ADDITIONAL FEE ($)
	Total (37 CFR 1.16(i))	*	Minus	**	=		x $0 =	
	Independent (37 CFR 1.16(h))	*	Minus	***	=		x $0 =	
	☐ Application Size Fee (37 CFR 1.16(s))							
	☐ FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))							
							TOTAL ADD'L FEE	

* If the entry in column 1 is less than the entry in column 2, write "0" in column 3.	LIE
** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, enter "20".	/ANGELA S WHITE/
*** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, enter "3".	
The "Highest Number Previously Paid For" (Total or Independent) is the highest number found in the appropriate box in column 1.	

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Inventor:	Jasmin Cosic	Confirmation No.: 1993
Title:	"ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION"	
Serial No.:	15/340,991	Filed: November 2, 2016
Examiner:	PARK, SOO JIN	Group Art Unit: 2668

Via EFS-Web May 27, 2019

Mail Stop Amendment
COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, VA 22313-1450

PRELIMINARY AMENDMENT

Dear Commissioner:

Before examination on the merits, please amend the claims in the above-identified application as shown in the following listing of claims. This preliminary amendment supersedes the preliminary amendment filed on May 4, 2019, therefore, the claims in the following listing of claims should be used for the examination on the merits instead of the claims in the preliminary amendment filed on May 4, 2019.

Electronic Acknowledgement Receipt

EFS ID:	36119055
Application Number:	15340991
International Application Number:	
Confirmation Number:	1993
Title of Invention:	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION
First Named Inventor/Applicant Name:	Jasmin Cosic
Customer Number:	116094
Filer:	Jasmin Cosic
Filer Authorized By:	
Attorney Docket Number:	
Receipt Date:	27-MAY-2019
Filing Date:	02-NOV-2016
Time Stamp:	19:58:52
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	no

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1		PRELIMINARY_AMENDMENT.pdf	47718 9af1a7f9b33fbe1848dd1e377d965a60824bd8f6	yes	13

Multipart Description/PDF files in .zip description		
Document Description	**Start**	**End**
Applicant Arguments/Remarks Made in an Amendment	13	13
Claims	2	12
Preliminary Amendment	1	1

Warnings:

Information:

Total Files Size (in bytes):	47718

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.
National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.
New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

Remarks

In this preliminary amendment, the applicant cancels claims 1-40, and presents for examination new claims 41-60. After entry of this preliminary amendment, claims 41-60 are pending. The undersigned may be contacted at (317) 772-1312 concerning this application.

I hereby certify that this correspondence is being submitted electronically via EFS Web to the United States Patent and Trademark Office.

By /Jasmin Cosic/
Jasmin Cosic

Date submitted: May 27, 2019

Respectfully submitted,

/Jasmin Cosic/

Jasmin Cosic

Listing of Claims

1 - 40 (Canceled)

41. (new) A method implemented using a computing system that includes one or more processor circuits, the method comprising:

accessing a memory that stores at least a first correlation including a first one or more object representations correlated with a first one or more instruction sets for operating a first device, wherein the first device includes a first one or more sensors, and wherein the first one or more object representations represent a first one or more objects detected at least in part by the first one or more sensors;

generating or receiving a second one or more object representations, wherein the second one or more object representations represent a second one or more objects;

determining the first one or more instruction sets for operating the first device based on at least partial match between the second one or more object representations and the first one or more object representations; and

at least in response to the determining, executing the first one or more instruction sets for operating the first device, wherein the first device or a second device performs one or more operations defined by the first one or more instruction sets for operating the first device.

42. (new) The method of claim 41, wherein at least one object of the first one or more objects and at least one object of the second one or more objects are the same.

43. (new) The method of claim 41, wherein the determining the first one or more instruction sets for operating the first device based on the at least partial match between the second one or more object representations and the first one or more object representations includes:

determining that a number of at least partially matching portions of the second one or more object representations and portions of the first one or more object representations exceeds a threshold number; or

determining that a percentage of at least partially matching portions of the second one or more object representations and portions of the first one or more object representations exceeds a threshold percentage.

44. (new) The method of claim 41, wherein the memory further stores at least a first knowledge cell, and wherein the first knowledge cell includes the first correlation.

45. (new) The method of claim 41, wherein the first one or more object representations include a first stream of one or more object representations, and

wherein the second one or more object representations include a second stream of one or more object representations.

46. (new) The method of claim 41, wherein, to correlate the first one or more object representations with the first one or more instruction sets for operating the first device, a determination is made that the first one or more instruction sets for operating the first device temporally correspond to the first one or more object representations.

47. (new) The method of claim 41, wherein elements of the computing system are included in: a single device or multiple devices, and wherein the one or more processor circuits include: one or more microcontrollers, one or more computing circuits, or one or more electronic circuits, and wherein the memory includes: a volatile memory or a non-volatile memory, and wherein the first device includes: a robot, a vehicle, an appliance, an electronic device, or a mechanical machine, and wherein the second device includes: a robot, a vehicle, an appliance, an electronic device, or a mechanical machine, and wherein the first one or more object representations include: one or more object properties of the first one or more objects, or one or more information on the first one or more objects, and wherein the second one or more object representations include: one or more object properties of the second one or more objects, or one or more information on the second one or more objects, and wherein an instruction set of the first one or more instruction sets for operating the first device includes at least one

selected from the group comprising: only one instruction, a plurality of instructions, one or more inputs, one or more commands, one or more computer commands, one or more keywords, one or more symbols, one or more operators, one or more variables, one or more values, one or more objects or object references, one or more data structures or data structure references, one or more functions or function references, one or more parameters, one or more signals, one or more characters, one or more digits, one or more user operating directions, one or more user directions, one or more user inputs, one or more representations of one or more user actions, one or more representations of one or more user clicks, one or more data, and one or more information, and wherein the first one or more sensors include at least one selected from the group comprising: one or more cameras, one or more microphones, one or more radars, one or more lidars, one or more sonars, and one or more apparatuses for detecting objects or object properties.

48. (new) The method of claim 41, wherein the memory further stores at least a third correlation including a third one or more object representations correlated with a third one or more instruction sets for operating the first device, and wherein the third one or more object representations represent a third one or more objects detected at least in part by the first one or more sensors, and wherein a first connection is generated to connect the first correlation with the third correlation, and wherein the first correlation connected with the third correlation form at least a portion of a knowledge structure or a knowledgebase

for operating at least one selected from the group comprising: the first device, the second device, and other one or more devices.

49. (new) The method of claim 41, wherein at least a portion of the first correlation is learned in a first learning process that includes:

generating or receiving the first one or more object representations; and

obtaining or receiving the first one or more instruction sets for operating the first device, and wherein the memory further stores at least a third correlation including a third one or more object representations correlated with a third one or more instruction sets for operating the second device, and wherein the second device includes a second one or more sensors, and wherein the third one or more object representations represent a third one or more objects detected at least in part by the second one or more sensors, and wherein at least a portion of the third correlation is learned in a second learning process that includes:

generating or receiving the third one or more object representations; and

obtaining or receiving the third one or more instruction sets for operating the second device.

50. (new) The method of claim 41, wherein the second one or more objects include one or more objects detected at least in part by: the first one or more sensors, or one or more sensors of the second device.

51. (new) The method of claim 41, further comprising:

modifying the first one or more instruction sets for operating the first device or a copy of the first one or more instruction sets for operating the first device, and wherein the executing the first one or more instruction sets for operating the first device includes executing the modified the first one or more instruction sets for operating the first device or the modified the copy of the first one or more instruction sets for operating the first device, and wherein the performing, by the first device or by the second device, the one or more operations defined by the first one or more instruction sets for operating the first device includes performing, by the first device or by the second device, the one or more operations defined by the modified the first one or more instruction sets for operating the first device or by the modified the copy of the first one or more instruction sets for operating the first device.

52. (new) The method of claim 41, wherein the first one or more object representations include or are associated with a first one or more extra information, and wherein the second one or more object representations include or are associated with a second one or more extra information, and wherein the determining the first one or more instruction sets for operating the first device based on the at least partial match between the second one or more object representations and the first one or more object representations is further based on at least partial match between the second one or more extra information and the first one or more extra information.

53. (new) The method of claim 41, wherein the memory further stores at least a third correlation including a third one or more object representations correlated with a third one or more instruction sets for operating the first device, and wherein the third one or more object representations represent a third one or more objects detected at least in part by the first one or more sensors, and wherein at least a portion of the first correlation is learned in a first learning process that includes operating the first device at least partially by a first user, and wherein at least a portion of the third correlation is learned in a second learning process that includes operating the first device at least partially by the first user.

54. (new) The method of claim 41, wherein the memory further stores at least a third correlation including a third one or more object representations correlated with a third one or more instruction sets for operating the first device, and wherein the third one or more object representations represent a third one or more objects detected at least in part by the first one or more sensors, and wherein at least a portion of the first correlation is learned in a first learning process that includes operating the first device at least partially by a first user, and wherein at least a portion of the third correlation is learned in a second learning process that includes operating the first device at least partially by a second user.

55. (new) The method of claim 41, wherein the memory further stores at least a third correlation including a third one or more object representations correlated with a third one or more instruction sets for operating the second device, and wherein the second device includes a second one or more sensors, and wherein the third one or more object representations represent a third one or more objects detected at least in part by the second one or more sensors, and wherein at least a portion of the first correlation is learned in a first learning process that includes operating the first device at least partially by a first user, and wherein at least a portion of the third correlation is learned in a second learning process that includes operating the second device at least partially by a second user.

56. (new) One or more non-transitory machine readable media storing machine readable code that when executed by one or more processor circuits causes the one or more processor circuits to perform at least:

accessing a memory that stores at least a first correlation including a first one or more object representations correlated with a first one or more instruction sets for operating a first device, wherein the first device includes a first one or more sensors, and wherein the first one or more object representations represent a first one or more objects detected at least in part by the first one or more sensors;

generating or receiving a second one or more object representations, wherein the second one or more object representations represent a second one or more objects;

determining the first one or more instruction sets for operating the first device based on at least partial match between the second one or more object representations and the first one or more object representations; and

at least in response to the determining, causing the first device or a second device to perform one or more operations defined by the first one or more instruction sets for operating the first device at least by causing the one or more processor circuits or another one or more processor circuits to execute the first one or more instruction sets for operating the first device.

57. (new) The one or more non-transitory machine readable media of claim 56, wherein, to correlate the first one or more object representations with the first one or more instruction sets for operating the first device, a determination is made that the first one or more instruction sets for operating the first device temporally correspond to the first one or more object representations.

58. (new) The one or more non-transitory machine readable media of claim 56, wherein the memory further stores at least a third correlation including a third one or more object representations correlated with a third one or more instruction sets for operating the first device, and wherein the third one or more object representations represent a third one or more objects detected at least in part by the first one or more sensors, and wherein at least a portion of the first correlation is learned in a first learning process that includes operating the first device at least partially by a first user, and wherein at least a portion of the third

correlation is learned in a second learning process that includes operating the first device at least partially by the first user.

59. (new) A system comprising:

a memory that stores at least a first correlation including a first one or more object representations correlated with a first one or more instruction sets for operating a first device, wherein the first device includes a first one or more sensors, and wherein the first one or more object representations represent a first one or more objects detected at least in part by the first one or more sensors;

a means for generating or receiving a second one or more object representations, wherein the second one or more object representations represent a second one or more objects;

a means for determining the first one or more instruction sets for operating the first device based on at least partial match between the second one or more object representations and the first one or more object representations; and

a means for executing the first one or more instruction sets for operating the first device at least in response to the determining, wherein the first device or a second device performs one or more operations defined by the first one or more instruction sets for operating the first device.

60. (new) The system of claim 59, wherein the means for generating or receiving the second one or more object representations includes one or more

processor circuits, and wherein the means for determining the first one or more instruction sets for operating the first device based on the at least partial match between the second one or more object representations and the first one or more object representations includes one or more processor circuits, and wherein the means for executing the first one or more instruction sets for operating the first device includes one or more processor circuits.

PTO/SB/06 (09-11)
Approved for use through 1/31/2014. OMB 0651-0032
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

PATENT APPLICATION FEE DETERMINATION RECORD Substitute for Form PTO-875	Application or Docket Number 15/340,991	Filing Date 11/02/2016	☐ To be Mailed

ENTITY: ☐ LARGE ☑ SMALL ☐ MICRO

APPLICATION AS FILED - PART I

FOR	(Column 1) NUMBER FILED	(Column 2) NUMBER EXTRA		RATE ($)	FEE ($)
☐ BASIC FEE (37 CFR 1.16(a), (b), or (c))	N/A	N/A		N/A	
☐ SEARCH FEE (37 CFR 1.16(k), (i), or (m))	N/A	N/A		N/A	
☐ EXAMINATION FEE (37 CFR 1.16(o), (p), or (q))	N/A	N/A		N/A	
TOTAL CLAIMS (37 CFR 1.16(i))	minus 20 =	*		x $40 =	
INDEPENDENT CLAIMS (37 CFR 1.16(h))	minus 3 =	*		x $210 =	
☐ APPLICATION SIZE FEE (37 CFR 1.16(s))	If the specification and drawings exceed 100 sheets of paper, the application size fee due is $310 ($155 for small entity) for each additional 50 sheets or fraction thereof. See 35 U.S.C. 41(a)(1)(G) and 37 CFR 1.16(s).				
☐ MULTIPLE DEPENDENT CLAIM PRESENT (37 CFR 1.16(j))					
* If the difference in column 1 is less than zero, enter "0" in column 2.				TOTAL	

APPLICATION AS AMENDED - PART II

		(Column 1)		(Column 2)	(Column 3)			
AMENDMENT	06/17/2019	CLAIMS REMAINING AFTER AMENDMENT		HIGHEST NUMBER PREVIOUSLY PAID FOR	PRESENT EXTRA		RATE ($)	ADDITIONAL FEE ($)
	Total (37 CFR 1.16(i))	* 20	Minus	** 20	= 0		x $50 =	0
	Independent (37 CFR 1.16(h))	* 3	Minus	*** 3	= 0		x $230 =	0
	☐ Application Size Fee (37 CFR 1.16(s))							
	☐ FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))							
							TOTAL ADD'L FEE	0

		(Column 1)		(Column 2)	(Column 3)			
AMENDMENT		CLAIMS REMAINING AFTER AMENDMENT		HIGHEST NUMBER PREVIOUSLY PAID FOR	PRESENT EXTRA		RATE ($)	ADDITIONAL FEE ($)
	Total (37 CFR 1.16(i))	*	Minus	**	=		x $0 =	
	Independent (37 CFR 1.16(h))	*	Minus	***	=		x $0 =	
	☐ Application Size Fee (37 CFR 1.16(s))							
	☐ FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))							
							TOTAL ADD'L FEE	

* If the entry in column 1 is less than the entry in column 2, write "0" in column 3.	LIE
** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, enter "20".	/ROLITA M WIMBUSH/
*** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, enter "3".	
The "Highest Number Previously Paid For" (Total or Independent) is the highest number found in the appropriate box in column 1.	

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Inventor:	Jasmin Cosic	Confirmation No.: 1993
Title:	"ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION"	
Serial No.:	15/340,991	Filed: November 2, 2016
Examiner:	GARCIA, SANTIAGO	Group Art Unit: 2668

Via EFS-Web June 17, 2019

Mail Stop Amendment
COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, VA 22313-1450

PRELIMINARY AMENDMENT

Dear Commissioner:

Before examination on the merits, please amend the claims in the above-identified application as shown in the following listing of claims. This preliminary amendment supersedes the preliminary amendments filed on May 4, 2019 and May 27, 2019, therefore, the claims in the following listing of claims should be used for the examination on the merits instead of the claims in the preliminary amendments filed on May 4, 2019 and May 27, 2019.

Listing of Claims

1 - 60 (canceled)

61. (new) A method implemented using a computing system that includes one or more processor circuits, the method comprising:

accessing a memory that stores at least a first correlation including a first one or more object representations correlated with a first one or more instruction sets for operating a first device, wherein the first one or more object representations represent a first one or more objects detected at least in part by one or more sensors of the first device;

generating a second one or more object representations, wherein the second one or more object representations represent a second one or more objects;

anticipating the first one or more instruction sets for operating the first device based on at least partial match between the second one or more object representations and the first one or more object representations; and

at least in response to the determining, executing the first one or more instruction sets for operating the first device, wherein the first device or a second device performs one or more operations defined by the first one or more instruction sets for operating the first device.

62. (new) The method of claim 61, wherein the executing the first one or more instruction sets for operating the first device is performed via an execution interface.

63. (new) The method of claim 61, wherein the first one or more instruction sets for operating the first device include one or more instruction sets for operating a portion of the first device, and wherein the performing, by the first device or by the second device, the one or more operations defined by the first one or more instruction sets for operating the first device includes performing, by the portion of the first device or by a portion of the second device, one or more operations defined by the one or more instruction sets for operating the portion of the first device.

64. (new) The method of claim 61, wherein the first one or more instruction sets for operating the first device are obtained via an acquisition interface.

65. (new) The method of claim 61, wherein at least one selected from the group comprising: the one or more processor circuits, and the memory are included in a remote computing device.

66. (new) The method of claim 61, wherein at least one selected from the group comprising: the one or more processor circuits, and the memory are included in multiple devices.

67. (new) The method of claim 61, wherein at least one selected from the group comprising: the one or more processor circuits, and the memory are included in a single device.

68. (new) The method of claim 61, wherein at least one object of the first one or more objects and at least one object of the second one or more objects are the same.

69. (new) The method of claim 61, wherein the second one or more objects include one or more objects detected at least in part by: the one or more sensors of the first device, or one or more sensors of the second device.

70. (new) The method of claim 61, wherein the one or more processor circuits include: one or more microcontrollers, one or more computing circuits, or one or more electronic circuits, and wherein the memory includes: a volatile memory or a non-volatile memory.

71. (new) One or more non-transitory machine readable media storing machine readable code that when executed by one or more processor circuits causes the one or more processor circuits to perform at least:

accessing a memory that stores at least a first correlation including a first one or more object representations correlated with a first one or more instruction

sets for operating a first device, wherein the first one or more object representations represent a first one or more objects detected at least in part by one or more sensors of the first device;

generating a second one or more object representations, wherein the second one or more object representations represent a second one or more objects;

determining the first one or more instruction sets for operating the first device based on at least partial match between the second one or more object representations and the first one or more object representations; and

at least in response to the determining, causing the first device or a second device to perform one or more operations defined by the first one or more instruction sets for operating the first device at least by causing the one or more processor circuits or another one or more processor circuits to execute the first one or more instruction sets for operating the first device.

72. (new) The one or more non-transitory machine readable media of claim 71, wherein the first one or more instruction sets for operating the first device include one or more instruction sets for operating a portion of the first device, and wherein the performing, by the first device or by the second device, the one or more operations defined by the first one or more instruction sets for operating the first device includes performing, by the portion of the first device or by a portion of the second device, one or more operations defined by the one or more instruction sets for operating the portion of the first device.

73. (new) The one or more non-transitory machine readable media of claim 71, wherein the first one or more instruction sets for operating the first device are obtained via an acquisition interface.

74. (new) The one or more non-transitory machine readable media of claim 71, wherein at least one selected from the group comprising: the one or more processor circuits, the another one or more processor circuits, and the memory are included in a remote computing device.

75. (new) The one or more non-transitory machine readable media of claim 71, wherein at least one selected from the group comprising: the one or more processor circuits, the another one or more processor circuits, and the memory are included in multiple devices.

76. (new) The one or more non-transitory machine readable media of claim 71, wherein at least one selected from the group comprising: the one or more processor circuits, the another one or more processor circuits, and the memory are included in a single device.

77. (new) The one or more non-transitory machine readable media of claim 71, wherein at least one object of the first one or more objects and at least one object of the second one or more objects are the same.

78. (new) The one or more non-transitory machine readable media of claim 71, wherein the second one or more objects include one or more objects detected at least in part by: the one or more sensors of the first device, or one or more sensors of the second device.

79. (new) A system comprising:

a memory that stores at least a first correlation including a first one or more object representations correlated with a first one or more instruction sets for operating a first device, wherein the first one or more object representations represent a first one or more objects detected at least in part by one or more sensors of the first device;

means for generating a second one or more object representations, wherein the second one or more object representations represent a second one or more objects;

means for determining the first one or more instruction sets for operating the first device based on at least partial match between the second one or more object representations and the first one or more object representations; and

means for executing the first one or more instruction sets for operating the first device at least in response to the determining, wherein the first device or a second device performs one or more operations defined by the first one or more instruction sets for operating the first device.

80. (new) The system of claim 79, wherein the means for generating the second one or more object representations includes one or more processor circuits, and wherein the means for determining the first one or more instruction sets for operating the first device based on the at least partial match between the second one or more object representations and the first one or more object representations includes one or more processor circuits, and wherein the means for executing the first one or more instruction sets for operating the first device includes one or more processor circuits.

Remarks

In this preliminary amendment, the applicant cancels claims 1-60, and presents for examination new claims 61-80. After entry of this preliminary amendment, claims 61-80 are pending. The undersigned may be contacted at (317) 772-1312 concerning this application.

I hereby certify that this correspondence is being submitted electronically via EFS Web to the United States Patent and Trademark Office.

By /Jasmin Cosic/
Jasmin Cosic

Date submitted: June 17, 2019

Respectfully submitted,

/Jasmin Cosic/

Jasmin Cosic

Electronic Acknowledgement Receipt

EFS ID:	36322513
Application Number:	15340991
International Application Number:	
Confirmation Number:	1993
Title of Invention:	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION
First Named Inventor/Applicant Name:	Jasmin Cosic
Customer Number:	116094
Filer:	Jasmin Cosic
Filer Authorized By:	
Attorney Docket Number:	
Receipt Date:	17-JUN-2019
Filing Date:	02-NOV-2016
Time Stamp:	17:21:14
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	no

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1		PRELIMINARY_AMENDMENT.pdf	34172 cc223f3bf2cb26a66d7d38330880f1f68ed85184	yes	9

	Multipart Description/PDF files in .zip description		
	Document Description	**Start**	**End**
	Preliminary Amendment	1	1
	Claims	2	8
	Applicant Arguments/Remarks Made in an Amendment	9	9
Warnings:			
Information:			
	Total Files Size (in bytes):	34172	

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.
National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.
New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

NOTICE OF ALLOWANCE AND FEE(S) DUE

116094 7590 06/24/2019

Jasmin Cosic
108 Woodbury Street
Pawtucket, RI 02861

EXAMINER	
GARCIA, SANTIAGO	
ART UNIT	PAPER NUMBER
2668	

DATE MAILED: 06/24/2019

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
15/340,991	11/02/2016	Jasmin Cosic		1993

TITLE OF INVENTION: ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION

APPLN. TYPE	ENTITY STATUS	ISSUE FEE DUE	PUBLICATION FEE DUE	PREV. PAID ISSUE FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	SMALL	$500	$0.00	$0.00	$500	09/24/2019

THE APPLICATION IDENTIFIED ABOVE HAS BEEN EXAMINED AND IS ALLOWED FOR ISSUANCE AS A PATENT. PROSECUTION ON THE MERITS IS CLOSED. THIS NOTICE OF ALLOWANCE IS NOT A GRANT OF PATENT RIGHTS. THIS APPLICATION IS SUBJECT TO WITHDRAWAL FROM ISSUE AT THE INITIATIVE OF THE OFFICE OR UPON PETITION BY THE APPLICANT. SEE 37 CFR 1.313 AND MPEP 1308.

THE ISSUE FEE AND PUBLICATION FEE (IF REQUIRED) MUST BE PAID WITHIN THREE MONTHS FROM THE MAILING DATE OF THIS NOTICE OR THIS APPLICATION SHALL BE REGARDED AS ABANDONED. THIS STATUTORY PERIOD CANNOT BE EXTENDED. SEE 35 U.S.C. 151. THE ISSUE FEE DUE INDICATED ABOVE DOES NOT REFLECT A CREDIT FOR ANY PREVIOUSLY PAID ISSUE FEE IN THIS APPLICATION. IF AN ISSUE FEE HAS PREVIOUSLY BEEN PAID IN THIS APPLICATION (AS SHOWN ABOVE), THE RETURN OF PART B OF THIS FORM WILL BE CONSIDERED A REQUEST TO REAPPLY THE PREVIOUSLY PAID ISSUE FEE TOWARD THE ISSUE FEE NOW DUE.

HOW TO REPLY TO THIS NOTICE:

I. Review the ENTITY STATUS shown above. If the ENTITY STATUS is shown as SMALL or MICRO, verify whether entitlement to that entity status still applies.

If the ENTITY STATUS is the same as shown above, pay the TOTAL FEE(S) DUE shown above.

If the ENTITY STATUS is changed from that shown above, on PART B - FEE(S) TRANSMITTAL, complete section number 5 titled "Change in Entity Status (from status indicated above)".

For purposes of this notice, small entity fees are 1/2 the amount of undiscounted fees, and micro entity fees are 1/2 the amount of small entity fees.

II. PART B - FEE(S) TRANSMITTAL, or its equivalent, must be completed and returned to the United States Patent and Trademark Office (USPTO) with your ISSUE FEE and PUBLICATION FEE (if required). If you are charging the fee(s) to your deposit account, section "4b" of Part B - Fee(s) Transmittal should be completed and an extra copy of the form should be submitted. If an equivalent of Part B is filed, a request to reapply a previously paid issue fee must be clearly made, and delays in processing may occur due to the difficulty in recognizing the paper as an equivalent of Part B.

III. All communications regarding this application must give the application number. Please direct all communications prior to issuance to Mail Stop ISSUE FEE unless advised to the contrary.

IMPORTANT REMINDER: Maintenance fees are due in utility patents issuing on applications filed on or after Dec. 12, 1980. It is patentee's responsibility to ensure timely payment of maintenance fees when due. More information is available at www.uspto.gov/PatentMaintenanceFees.

PART B - FEE(S) TRANSMITTAL

Complete and send this form, together with applicable fee(s), by mail or fax, or via EFS-Web.

By mail, send to: Mail Stop ISSUE FEE
Commissioner for Patents
P.O. Box 1450
Alexandria, Virginia 22313-1450

By fax, send to: (571)-273-2885

INSTRUCTIONS: This form should be used for transmitting the ISSUE FEE and PUBLICATION FEE (if required). Blocks 1 through 5 should be completed where appropriate. All further correspondence including the Patent, advance orders and notification of maintenance fees will be mailed to the current correspondence address as indicated unless corrected below or directed otherwise in Block 1, by (a) specifying a new correspondence address; and/or (b) indicating a separate "FEE ADDRESS" for maintenance fee notifications.

CURRENT CORRESPONDENCE ADDRESS (Note: Use Block 1 for any change of address)

116094 7590 06/24/2019

Jasmin Cosic
108 Woodbury Street
Pawtucket, RI 02861

Note: A certificate of mailing can only be used for domestic mailings of the Fee(s) Transmittal. This certificate cannot be used for any other accompanying papers. Each additional paper, such as an assignment or formal drawing, must have its own certificate of mailing or transmission.

Certificate of Mailing or Transmission

I hereby certify that this Fee(s) Transmittal is being deposited with the United States Postal Service with sufficient postage for first class mail in an envelope addressed to the Mail Stop ISSUE FEE address above, or being transmitted to the USPTO via EFS-Web or by facsimile to (571) 273-2885, on the date below.

(Typed or printed name)

(Signature)

(Date)

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
15/340,991	11/02/2016	Jasmin Cosic		1993

TITLE OF INVENTION: ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION

APPLN. TYPE	ENTITY STATUS	ISSUE FEE DUE	PUBLICATION FEE DUE	PREV. PAID ISSUE FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	SMALL	$500	$0.00	$0.00	$500	09/24/2019

EXAMINER	ART UNIT	CLASS-SUBCLASS
GARCIA, SANTIAGO	2668	382-155000

1. Change of correspondence address or indication of "Fee Address" (37 CFR 1.363).

❑ Change of correspondence address (or Change of Correspondence Address form PTO/SB/122) attached.

❑ "Fee Address" indication (or "Fee Address" Indication form PTO/SB/47; Rev 03-09 or more recent) attached. **Use of a Customer Number is required.**

2. For printing on the patent front page, list

(1) The names of up to 3 registered patent attorneys or agents OR, alternatively,

(2) The name of a single firm (having as a member a registered attorney or agent) and the names of up to 2 registered patent attorneys or agents. If no name is listed, no name will be printed.

1________________

2________________

3________________

3. ASSIGNEE NAME AND RESIDENCE DATA TO BE PRINTED ON THE PATENT (print or type)

PLEASE NOTE: Unless an assignee is identified below, no assignee data will appear on the patent. If an assignee is identified below, the document must have been previously recorded, or filed for recordation, as set forth in 37 CFR 3.11 and 37 CFR 3.81(a). Completion of this form is NOT a substitute for filing an assignment.

(A) NAME OF ASSIGNEE

(B) RESIDENCE: (CITY and STATE OR COUNTRY)

Please check the appropriate assignee category or categories (will not be printed on the patent) : ❑ Individual ❑ Corporation or other private group entity ❑ Government

4a. Fees submitted: ❑Issue Fee ❑Publication Fee (if required) ❑Advance Order - # of Copies __________

4b. Method of Payment: *(Please first reapply any previously paid fee shown above)*

❑ Electronic Payment via EFS-Web ❑ Enclosed check ❑ Non-electronic payment by credit card (Attach form PTO-2038)

❑ The Director is hereby authorized to charge the required fee(s), any deficiency, or credit any overpayment to Deposit Account No. __________

5. **Change in Entity Status** (from status indicated above)

❑ Applicant certifying micro entity status. See 37 CFR 1.29

❑ Applicant asserting small entity status. See 37 CFR 1.27

❑ Applicant changing to regular undiscounted fee status.

NOTE: Absent a valid certification of Micro Entity Status (see forms PTO/SB/15A and 15B), issue fee payment in the micro entity amount will not be accepted at the risk of application abandonment.
NOTE: If the application was previously under micro entity status, checking this box will be taken to be a notification of loss of entitlement to micro entity status.
NOTE: Checking this box will be taken to be a notification of loss of entitlement to small or micro entity status, as applicable.

NOTE: This form must be signed in accordance with 37 CFR 1.31 and 1.33. See 37 CFR 1.4 for signature requirements and certifications.

Authorized Signature ________________ Date ________________

Typed or printed name ________________ Registration No. ________________

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
15/340,991	11/02/2016	Jasmin Cosic		1993

116094 7590 06/24/2019

Jasmin Cosic
108 Woodbury Street
Pawtucket, RI 02861

EXAMINER	
GARCIA, SANTIAGO	
ART UNIT	PAPER NUMBER
2668	

DATE MAILED: 06/24/2019

Determination of Patent Term Adjustment under 35 U.S.C. 154 (b)

(Applications filed on or after May 29, 2000)

The Office has discontinued providing a Patent Term Adjustment (PTA) calculation with the Notice of Allowance.

Section 1(h)(2) of the AIA Technical Corrections Act amended 35 U.S.C. 154(b)(3)(B)(i) to eliminate the requirement that the Office provide a patent term adjustment determination with the notice of allowance. See Revisions to Patent Term Adjustment, 78 Fed. Reg. 19416, 19417 (Apr. 1, 2013). Therefore, the Office is no longer providing an initial patent term adjustment determination with the notice of allowance. The Office will continue to provide a patent term adjustment determination with the Issue Notification Letter that is mailed to applicant approximately three weeks prior to the issue date of the patent, and will include the patent term adjustment on the patent. Any request for reconsideration of the patent term adjustment determination (or reinstatement of patent term adjustment) should follow the process outlined in 37 CFR 1.705.

Any questions regarding the Patent Term Extension or Adjustment determination should be directed to the Office of Patent Legal Administration at (571)-272-7702. Questions relating to issue and publication fee payments should be directed to the Customer Service Center of the Office of Patent Publication at 1-(888)-786-0101 or (571)-272-4200.

OMB Clearance and PRA Burden Statement for PTOL-85 Part B

The Paperwork Reduction Act (PRA) of 1995 requires Federal agencies to obtain Office of Management and Budget approval before requesting most types of information from the public. When OMB approves an agency request to collect information from the public, OMB (i) provides a valid OMB Control Number and expiration date for the agency to display on the instrument that will be used to collect the information and (ii) requires the agency to inform the public about the OMB Control Number's legal significance in accordance with 5 CFR 1320.5(b).

The information collected by PTOL-85 Part B is required by 37 CFR 1.311. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 30 minutes to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, Virginia 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, Virginia 22313-1450. Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether disclosure of these records is required by the Freedom of Information Act.
2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.
3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.
4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).
5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.
6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).
7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.
8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspection or an issued patent.
9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Notice of Allowability	**Application No.** 15/340,991	**Applicant(s)** Cosic, Jasmin	
	Examiner SANTIAGO GARCIA	**Art Unit** 2668	**AIA (FITF) Status** Yes

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address--

All claims being allowable, PROSECUTION ON THE MERITS IS (OR REMAINS) CLOSED in this application. If not included herewith (or previously mailed), a Notice of Allowance (PTOL-85) or other appropriate communication will be mailed in due course. **THIS NOTICE OF ALLOWABILITY IS NOT A GRANT OF PATENT RIGHTS.** This application is subject to withdrawal from issue at the initiative of the Office or upon petition by the applicant. See 37 CFR 1.313 and MPEP 1308.

1.☑ This communication is responsive to 06/17/19 last set of claims submitted.
☐ A declaration(s)/affidavit(s) under **37 CFR 1.130(b)** was/were filed on ______.

2.☐ An election was made by the applicant in response to a restriction requirement set forth during the interview on ______; the restriction requirement and election have been incorporated into this action.

3.☑ The allowed claim(s) is/are 61-80 . As a result of the allowed claim(s), you may be eligible to benefit from the **Patent Prosecution Highway** program at a participating intellectual property office for the corresponding application. For more information, please see **http://www.uspto.gov/patents/init_events/pph/index.jsp** or send an inquiry to **PPHfeedback@uspto.gov.**

4.☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

Certified copies:

a) ☐All b) ☐ Some *c) ☐ None of the:

1. ☐ Certified copies of the priority documents have been received.
2. ☐ Certified copies of the priority documents have been received in Application No. ______ .
3. ☐ Copies of the certified copies of the priority documents have been received in this national stage application from the International Bureau (PCT Rule 17.2(a)).

* Certified copies not received: ______ .

Applicant has THREE MONTHS FROM THE "MAILING DATE" of this communication to file areply complying with the requirements noted below. Failure to timely comply will result in ABANDONMENT of this application.
THIS THREE-MONTH PERIOD IS NOT EXTENDABLE.

5.☐ CORRECTED DRAWINGS (as "replacement sheets") must be submitted.
☐ including changes required by the attached Examiner's Amendment / Comment or in the Office action of Paper No./Mail Date ______ .

Identifying indicia such as the application number (see 37 CFR 1.84(c)) should be written on the drawings in the front (not the back) of each sheet. Replacement sheet(s) should be labeled as such in the header according to 37 CFR 1.121(d).

6.☐ DEPOSIT OF and/or INFORMATION about the deposit of BIOLOGICAL MATERIAL must be submitted. Note the attached Examiner's comment regarding REQUIREMENT FOR THE DEPOSIT OF BIOLOGICAL MATERIAL.

Attachment(s)

1.☑ Notice of References Cited (PTO-892)
2.☑ Information Disclosure Statements (PTO/SB/08), Paper No./Mail Date ______.
3.☐ Examiner's Comment Regarding Requirement for Deposit of Biological Material ______.
4.☐ Interview Summary (PTO-413), Paper No./Mail Date. ______.
5. ☑ Examiner's Amendment/Comment
6. ☑ Examiner's Statement of Reasons for Allowance
7. ☐ Other ______.

/SANTIAGO GARCIA/
Primary Examiner, Art Unit 2668

Notice of Pre-AIA or AIA Status

The present application, filed on or after March 16, 2013, is being examined under the first inventor to file provisions of the AIA.

Claim Interpretation

The following is a quotation of 35 U.S.C. 112(f):

> (f) Element in Claim for a Combination. – An element in a claim for a combination may be expressed as a means or step for performing a specified function without the recital of structure, material, or acts in support thereof, and such claim shall be construed to cover the corresponding structure, material, or acts described in the specification and equivalents thereof.

The following is a quotation of pre-AIA 35 U.S.C. 112, sixth paragraph:

> An element in a claim for a combination may be expressed as a means or step for performing a specified function without the recital of structure, material, or acts in support thereof, and such claim shall be construed to cover the corresponding structure, material, or acts described in the specification and equivalents thereof.

The claims in this application are given their broadest reasonable interpretation using the plain meaning of the claim language in light of the specification as it would be understood by one of ordinary skill in the art. The broadest reasonable interpretation of a claim element (also commonly referred to as a claim limitation) is limited by the description in the specification when 35 U.S.C. 112(f) or pre-AIA 35 U.S.C. 112, sixth paragraph, is invoked.

As explained in MPEP § 2181, subsection I, claim limitations that meet the following three-prong test will be interpreted under 35 U.S.C. 112(f) or pre-AIA 35 U.S.C. 112, sixth paragraph:

(A) the claim limitation uses the term "means" or "step" or a term used as a substitute for "means" that is a generic placeholder (also called a nonce term or a non-structural term having no specific structural meaning) for performing the claimed function;

(B) the term “means” or “step” or the generic placeholder is modified by functional language, typically, but not always linked by the transition word “for” (e.g., “means for”) or another linking word or phrase, such as “configured to” or “so that”; and

(C) the term “means” or “step” or the generic placeholder is not modified by sufficient structure, material, or acts for performing the claimed function.

Use of the word “means” (or “step”) in a claim with functional language creates a rebuttable presumption that the claim limitation is to be treated in accordance with 35 U.S.C. 112(f) or pre-AIA 35 U.S.C. 112, sixth paragraph. The presumption that the claim limitation is interpreted under 35 U.S.C. 112(f) or pre-AIA 35 U.S.C. 112, sixth paragraph, is rebutted when the claim limitation recites sufficient structure, material, or acts to entirely perform the recited function.

Absence of the word “means” (or “step”) in a claim creates a rebuttable presumption that the claim limitation is not to be treated in accordance with 35 U.S.C. 112(f) or pre-AIA 35 U.S.C. 112, sixth paragraph. The presumption that the claim limitation is not interpreted under 35 U.S.C. 112(f) or pre-AIA 35 U.S.C. 112, sixth paragraph, is rebutted when the claim limitation recites function without reciting sufficient structure, material or acts to entirely perform the recited function.

Claim limitations 79-80 in this application that use the word “means” (or “step”) are being interpreted under 35 U.S.C. 112(f) or pre-AIA 35 U.S.C. 112, sixth paragraph, except as otherwise indicated in an Office action. Conversely, claim limitations in this application that do not use the word “means” (or “step”) are not being interpreted under 35 U.S.C. 112(f) or pre-AIA 35 U.S.C. 112, sixth paragraph, except as otherwise indicated in an Office action.

EXAMINER'S AMENDMENT

An examiner's amendment to the record appears below. Should the changes and/or additions be unacceptable to applicant, an amendment may be filed as provided by 37 CFR 1.312. To ensure consideration of such an amendment, it MUST be submitted no later than the payment of the issue fee.

Authorization for this examiner's amendment was given in an interview with Jasmin, Cosic on 6/18/19.

All claims 61-80 are amended as follows, all claims are currently amended.

The application has been amended as follows:

61. A method implemented using a computing system that includes one or more processor circuits, the method comprising:

accessing a memory that stores at least a knowledgebase that includes: a first correlation including a first circumstance representation correlated with a first one or more instruction sets for operating a first device and a second correlation including a second circumstance representation correlated with a second one or more instruction sets for operating the first device, wherein the first circumstance representation represents a first circumstance detected at least in part by one or more sensors of the first device and the second circumstance representation represents a second circumstance detected at least in part by the one or more sensors of the first device, and wherein at least a portion of the first correlation and at least a portion of the second correlation are learned in a learning process that includes operating the first device at least partially by a user;

generating or receiving a third circumstance representation, wherein the third circumstance representation represents a third circumstance detected at least in part by the one or more sensors of the first device or at least in part by one or more sensors of a second device;

anticipating the first one or more instruction sets for operating the first device learned in the learning process based on at least partial match between the third circumstance representation and the first circumstance representation; and

at least in response to the anticipating, executing the first one or more instruction sets for operating the first device learned in the learning process, wherein the first device or [[a]] the second device autonomously performs one or more operations defined by the first one or more instruction sets for operating the first device learned in the learning process.

62. The method of claim 61, wherein the knowledgebase further includes: a first knowledge cell and a second knowledge cell, and wherein the first knowledge cell includes the first correlation and the second knowledge cell includes the second correlation.

63. The method of claim 61, wherein the learning process includes: creating, inserting, deleting, modifying, or manipulating an element of the first correlation, or creating, inserting, deleting, modifying, or manipulating an element of the second correlation.

64. The method of claim 61, wherein the learning process includes:

generating or receiving the first circumstance representation, and generating or receiving the second circumstance representation; and

obtaining or receiving the first one or more instruction sets for operating the first device, and obtaining or receiving the second one or more instruction sets for operating the first device, and wherein the knowledgebase further includes a fourth correlation including a fourth circumstance representation correlated with a fourth one or more instruction sets for operating the second device, and wherein the fourth circumstance representation represents a fourth circumstance detected at least

in part by the one or more sensors of the second device, and wherein at least a portion of the fourth correlation is learned in another learning process that includes:

generating or receiving the fourth circumstance representation; and

obtaining or receiving the fourth one or more instruction sets for operating the second device.

65. The method of claim 61, wherein the anticipating the first one or more instruction sets for operating the first device learned in the learning process based on the at least partial match between the third circumstance representation and the first circumstance representation includes:

determining that a number of at least partially matching portions of the third circumstance representation and portions of the first circumstance representation exceeds a threshold number; or

determining that a percentage of at least partially matching portions of the third circumstance representation and portions of the first circumstance representation exceeds a threshold percentage.

66. The method of claim 61, wherein the at least the portion of the first correlation and the at least the portion of the second correlation are learned in the learning process while the user operates the first device, and wherein the one or more operations defined by the first one or more instruction sets for operating the first device learned in the learning process correspond to the user's methodology of operating the first device learned in the learning process.

67. The method of claim 61, wherein the knowledgebase further includes a fourth correlation including a fourth circumstance representation correlated with a fourth one or more instruction sets for operating: the first device, the second device, or a third device, and wherein a first connection is generated to connect the first correlation with the second correlation, and wherein a second connection is generated to connect the second correlation with the fourth correlation.

68. The method of claim 61, further comprising:

modifying the first one or more instruction sets for operating the first device learned in the learning process or a copy of the first one or more instruction sets for operating the first device learned in the learning process, and wherein the executing the first one or more instruction sets for operating the first device learned in the learning process includes executing the modified the first one or more instruction sets for operating the first device learned in the learning process or the modified the copy of the first one or more instruction sets for operating the first device learned in the learning process, and wherein the performing, by the first device or by the second device, the one or more operations defined by the first one or more instruction sets for operating the first device learned in the learning process includes performing, by the first device or by the second device, one or more operations defined by the modified the first one or more instruction sets for operating the first device learned in the learning process or by the modified the copy of the first one or more instruction sets for operating the first device learned in the learning process.

69. The method of claim 61, wherein the knowledgebase further includes a fourth correlation including a fourth circumstance representation correlated with a fourth one or more instruction sets for operating the first device, and wherein the fourth circumstance representation represents a fourth circumstance detected at least in part by the one or more sensors of the first device, and wherein at

least a portion of the fourth correlation is learned in another learning process that includes operating the first device at least partially by the user.

70. The method of claim 61, wherein the knowledgebase further includes a fourth correlation including a fourth circumstance representation correlated with a fourth one or more instruction sets for operating the first device, and wherein the fourth circumstance representation represents a fourth circumstance detected at least in part by the one or more sensors of the first device, and wherein at least a portion of the fourth correlation is learned in another learning process that includes operating the first device at least partially by another user.

71. The method of claim 61, wherein the knowledgebase further includes a fourth correlation including a fourth circumstance representation correlated with a fourth one or more instruction sets for operating the second device, and wherein the fourth circumstance representation represents a fourth circumstance detected at least in part by the one or more sensors of the second device, and wherein at least a portion of the fourth correlation is learned in another learning process that includes operating the second device at least partially by another user.

72. The method of claim 61, wherein the first circumstance includes one or more objects detected at least in part by the one or more sensors of the first device at a first time or during a first time period, and wherein the second circumstance includes one or more objects detected at least in part by the one or more sensors of the first device at a second time or during a second time period, and wherein the third circumstance includes:

one or more objects detected at least in part by the one or more sensors of the first device at a third time or during a third time period, or

one or more objects detected at least in part by the one or more sensors of the second device at a third time or during a third time period.

73. The method of claim 61, wherein the first circumstance representation is a data structure including one or more data about the first circumstance of the first device, and wherein the second circumstance representation is a data structure including one or more data about the second circumstance of the first device, and wherein the third circumstance representation is a data structure including one or more data about: the third circumstance of the first device or the third circumstance of the second device.

74. The method of claim 61, wherein the first circumstance representation includes: one or more object representations, or one or more collections of object representations, and wherein the second circumstance representation includes: one or more object representations, or one or more collections of object representations, and wherein the third circumstance representation includes: one or more object representations, or one or more collections of object representations.

75. The method of claim 61, wherein, to correlate the first circumstance representation with the first one or more instruction sets for operating the first device, a determination is made that the first one or more instruction sets for operating the first device temporally correspond to the first circumstance representation, wherein the first circumstance representation includes: one or more object representations, or one or more collections of object representations.

76. The method of claim 61, wherein the anticipating the first one or more instruction sets for operating the first device learned in the learning process based on the at least partial match between

the third circumstance representation and the first circumstance representation includes determining the first one or more instruction sets for operating the first device learned in the learning process based on the at least partial match between the third circumstance representation and the first circumstance representation.

77. The method of claim 61, wherein elements of the computing system are included in: a single device, or multiple devices, and wherein the one or more processor circuits include: one or more microcontrollers, one or more computing circuits, or one or more electronic circuits, and wherein the memory includes: a volatile memory, or a non-volatile memory, and wherein the first device includes: a robot, a vehicle, an appliance, an electronic device, or a mechanical machine, and wherein the second device includes: a robot, a vehicle, an appliance, an electronic device, or a mechanical machine, and wherein an instruction set of the first one or more instruction sets for operating the first device includes at least one selected from the group comprising: only one instruction, a plurality of instructions, one or more inputs, one or more commands, one or more computer commands, one or more keywords, one or more symbols, one or more operators, one or more variables, one or more values, one or more objects or object references, one or more data structures or data structure references, one or more functions or function references, one or more parameters, one or more signals, one or more characters, one or more digits, one or more user operating directions, one or more user directions, one or more user inputs, one or more representations of one or more user actions, one or more representations of one or more user clicks, one or more binary bits, one or more assembly language commands, one or more states, one or more codes, one or more data, and one or more information, and wherein an instruction set of the second one or more instruction sets for operating the first device includes at least one selected from the group comprising: only one instruction, a plurality of instructions, one or more inputs, one or more commands, one or more computer commands, one or more keywords, one or more symbols, one or

more operators, one or more variables, one or more values, one or more objects or object references, one or more data structures or data structure references, one or more functions or function references, one or more parameters, one or more signals, one or more characters, one or more digits, one or more user operating directions, one or more user directions, one or more user inputs, one or more representations of one or more user actions, one or more representations of one or more user clicks, one or more binary bits, one or more assembly language commands, one or more states, one or more codes, one or more data, and one or more information, and wherein the one or more sensors of the first device include at least one selected from the group comprising: one or more cameras, one or more microphones, one or more radars, one or more lidars, one or more sonars, and one or more apparatuses for detecting objects or object properties, and wherein the one or more sensors of the second device include at least one selected from the group comprising: one or more cameras, one or more microphones, one or more radars, one or more lidars, one or more sonars, and one or more apparatuses for detecting objects or object properties, and wherein the at least the portion of the first correlation includes: one portion of the first correlation, multiple portions of the first correlation, all portions of the first correlation, or the entire first correlation, and wherein the at least the portion of the second correlation includes: one portion of the second correlation, multiple portions of the second correlation, all portions of the second correlation, or the entire second correlation, and wherein an object of the first circumstance is the same as an object of the third circumstance, or multiple objects of the first circumstance are the same as multiple objects of the third circumstance, or all objects of the first circumstance are the same as all objects of the third circumstance, or all objects of the first circumstance are different than all objects of the third circumstance.

78. One or more non-transitory machine readable media storing machine readable code that when executed by one or more processor circuits causes the one or more processor circuits to perform at least:

accessing a memory that stores at least a knowledgebase that includes: a first correlation including a first circumstance representation correlated with a first one or more instruction sets for operating a first device and a second correlation including a second circumstance representation correlated with a second one or more instruction sets for operating the first device, wherein the first circumstance representation represents a first circumstance detected at least in part by one or more sensors of the first device and the second circumstance representation represents a second circumstance detected at least in part by the one or more sensors of the first device, and wherein at least a portion of the first correlation and at least a portion of the second correlation are learned in a learning process that includes operating the first device at least partially by a user;

generating or receiving a third circumstance representation, wherein the third circumstance representation represents a third circumstance detected at least in part by the one or more sensors of the first device or at least in part by one or more sensors of a second device;

anticipating the first one or more instruction sets for operating the first device learned in the learning process based on at least partial match between the third circumstance representation and the first circumstance representation; and

at least in response to the anticipating, causing the first device or [[a]] the second device to perform one or more operations defined by the first one or more instruction sets for operating the first device learned in the learning process at least by causing the one or more processor circuits or another one or more processor circuits to execute the first one or more instruction sets for operating the first device learned in the learning process.

79. A system comprising:

a memory that stores at least a knowledgebase that includes: a first correlation including a first circumstance representation correlated with a first one or more instruction sets for operating a first device and a second correlation including a second circumstance representation correlated with a second one or more instruction sets for operating the first device, wherein the first circumstance representation represents a first circumstance detected at least in part by one or more sensors of the first device and the second circumstance representation represents a second circumstance detected at least in part by the one or more sensors of the first device, and wherein at least a portion of the first correlation and at least a portion of the second correlation are learned in a learning process that includes operating the first device at least partially by a user;

means for generating or receiving a third circumstance representation, wherein the third circumstance representation represents a third circumstance detected at least in part by the one or more sensors of the first device or at least in part by one or more sensors of a second device;

means for anticipating the first one or more instruction sets for operating the first device learned in the learning process based on at least partial match between the third circumstance representation and the first circumstance representation; and

means for executing, at least in response to the anticipating, the first one or more instruction sets for operating the first device learned in the learning process, wherein the first device or [[a]] the second device autonomously performs one or more operations defined by the first one or more instruction sets for operating the first device learned in the learning process.

80. The system of claim 79, wherein the means for generating or receiving the third circumstance representation includes one or more processor circuits, and wherein the means for anticipating the first one or more instruction sets for operating the first device learned in the learning

process based on the at least partial match between the third circumstance representation and the first circumstance representation includes one or more processor circuits, and wherein the means for executing, at least in response to the anticipating, the first one or more instruction sets for operating the first device learned in the learning process includes one or more processor circuits.

Reasons for Allowance

The following is an examiner's statement of reasons for allowance: Applicant has disclosed an artificial intelligence system that creates a knowledge base based on operations happening by a device and partially used by a user in part. After instruction sets get stored in a training mode by matching up an instances or objects with training sets, correlations happen as signals from sensors start to come in as a devices gets used. These correlations correlate instructions sets that were gathered in the training mode and stored in the knowledge base and the system then expects different actions according to the instruction sets to happen based on these correlations which in part are caused by a user using the device. Then automatically in response to the anticipating, executing the first one or more instruction sets for operating the first device learned in the learning process, wherein the first device or a second device autonomously performs one or more operations defined by the first one or more instruction sets for operating the first device learned in the learning process and that come up in the correlating process and the anticipating step. There are references that claim the broad spectrum idea, however they do not deal with the specifics of the current claim limitations, such as the prior art does not disclose the device that operates as the user directs it, hence, does not disclose learning the device operation from the user directing the device. Prior art discloses a system that explores its own environment on its own, which is very different from relying on the user to direct it. Prior art does not disclose the first and the second correlations that each include a circumstance representation correlated with instruction sets. Prior art

does not disclose the knowledgebase, which includes a 3-level nested data structure recited in the independent claims. Such as individual circumstance representation (a data structure itself) and individual instruction set (a data structure itself when learned in the learning process). And correlation (a data structure itself) that includes a circumstance representation correlated with instruction sets. And knowledgebase a data structure itself including a plurality of correlation data structures in various arrangements. Further autonomous operation of the device in the claims performs the operation and does not rely on the re-training such as in the prior art as noted below, and the re-training is a crucial function in the proper operating of the systems in the prior art. As an example for highlighting these differences Schnittman (US 2016/0167226) in fig.5 505-510 can be consider the training, and determine robot behavior 535 can be consider instruction sets, and 545 can be considered a correlation since it is going to grab images from the buffer and change the database accordingly, then the classifier can be considered the instances. However the automatic portion of the device actually performing the task without the re-training based on those correlations is not in the prior art as an example. There is always a re-training of some sorts, where the current application automatically controls the device based on that prior correlation and no need for re-training of any kind. Many other references could be cited but the same analysis and differences will be noted.

Any comments considered necessary by applicant must be submitted no later than the payment of the issue fee and, to avoid processing delays, should preferably accompany the issue fee. Such submissions should be clearly labeled “Comments on Statement of Reasons for Allowance.”

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to SANTIAGO GARCIA whose telephone number is (571)270-5182. The examiner can normally be reached on Monday-Friday 9:30am-5:30pm.

Examiner interviews are available via telephone, in-person, and video conferencing using a USPTO supplied web-based collaboration tool. To schedule an interview, applicant is encouraged to use the USPTO Automated Interview Request (AIR) at http://www.uspto.gov/interviewpractice.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, **Le, Vu** can be reached on **(571) 272-7332**. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/SANTIAGO GARCIA/
Primary Examiner, Art Unit 2668

/SG/

Notice of References Cited	Application/Control No. 15/340,991	Applicant(s)/Patent Under Reexamination Cosic, Jasmin	
	Examiner SANTIAGO GARCIA	Art Unit 2668	Page 1 of 2

U.S. PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Name	CPC Classification	US Classification
*	A	US-20160243701-A1	08-2016	Gildert; Suzanne	G05B19/42	1/1
*	B	US-20180120813-A1	05-2018	COFFMAN; Valerie R.	G05B19/4097	1/1
*	C	US-20180101146-A1	04-2018	Hariharan; Rajaraman	G05B13/0265	1/1
*	D	US-8595154-B2	11-2013	Breckenridge; Jordan M.	G06N20/00	706/12
*	E	US-20160274187-A1	09-2016	MENON; SANKARAN M.	G01R31/31718	1/1
*	F	US-9367806-B1	06-2016	Cosic; Jasmin	G06N5/022	1/1
*	G	US-20160167226-A1	06-2016	Schnittman; Mark S.	B25J5/00	382/153
*	H	US-9282309-B1	03-2016	Cosic; Jasmin	H04N9/87	1/1
*	I	US-20150339213-A1	11-2015	LEE; Christopher Stephen	G06F11/3664	717/125
*	J	US-20140177946-A1	06-2014	LIM; Kil-Taek	G06K9/4614	382/156
*	K	US-20120290347-A1	11-2012	ELAZOUNI; ASHRAF	G06Q10/00	705/7.12
*	L	US-20120167057-A1	06-2012	Schmich; Christopher P.	G06F11/3644	717/130
*	M	US-20100278420-A1	11-2010	Shet; Vinay Damodar	G06K9/00369	382/156

FOREIGN PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Country	Name	CPC Classification
	N					
	O					
	P					
	Q					
	R					
	S					
	T					

NON-PATENT DOCUMENTS

*		Include as applicable: Author, Title Date, Publisher, Edition or Volume, Pertinent Pages)
	U	
	V	
	W	
	X	

*A copy of this reference is not being furnished with this Office action. (See MPEP § 707.05(a).)
Dates in MM-YYYY format are publication dates. Classifications may be US or foreign.

Notice of References Cited	Application/Control No. 15/340,991	Applicant(s)/Patent Under Reexamination Cosic, Jasmin	
	Examiner SANTIAGO GARCIA	Art Unit 2668	Page 2 of 2

U.S. PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Name	CPC Classification	US Classification
*	A	US-20170161958-A1	06-2017	EILAT; Eran	G02B27/017	1/1
*	B	US-20160077166-A1	03-2016	Morozov; Alexey	G01C19/00	702/150
*	C	US-9443192-B1	09-2016	Cosic; Jasmin	G06N3/02	1/1
*	D	US-6408262-B1	06-2002	Leerberg; Henrik	G06F11/3608	703/2
*	E	US-9864933-B1	01-2018	Cosic; Jasmin	G06K9/66	1/1
*	F	US-10102449-B1	10-2018	Cosic; Jasmin	G06K9/00771	1/1
*	G	US-10102226-B1	10-2018	Cosic; Jasmin	G06F16/5866	1/1
*	H	US-9582762-B1	02-2017	Cosic; Jasmin	G06N5/04	1/1
	I					
	J					
	K					
	L					
	M					

FOREIGN PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Country	Name	CPC Classification
	N					
	O					
	P					
	Q					
	R					
	S					
	T					

NON-PATENT DOCUMENTS

*		Include as applicable: Author, Title Date, Publisher, Edition or Volume, Pertinent Pages)
	U	
	V	
	W	
	X	

*A copy of this reference is not being furnished with this Office action. (See MPEP § 707.05(a).)
Dates in MM-YYYY format are publication dates. Classifications may be US or foreign.

Index of Claims	Application/Control No. 15/340,991	Applicant(s)/Patent Under Reexamination Cosic, Jasmin
	Examiner SANTIAGO GARCIA	Art Unit 2668

✓	Rejected	-	Cancelled	N	Non-Elected	A	Appeal
=	Allowed	÷	Restricted	I	Interference	O	Objected

CLAIMS

☐ Claims renumbered in the same order as presented by applicant ☐ CPA ☐ T.D. ☐ R.1.47

CLAIM		DATE								
Final	Original	06/19/2019								
	1	-								
	2	-								
	3	-								
	4	-								
	5	-								
	6	-								
	7	-								
	8	-								
	9	-								
	10	-								
	11	-								
	12	-								
	13	-								
	14	-								
	15	-								
	16	-								
	17	-								
	18	-								
	19	-								
	20	-								
	21	-								
	22	-								
	23	-								
	24	-								
	25	-								
	26	-								
	27	-								
	28	-								
	29	-								
	30	-								
	31	-								
	32	-								
	33	-								
	34	-								
	35	-								
	36	-								
	37	-								
	38	-								
	39	-								
	40	-								
	41	-								
	42	-								

Index of Claims	Application/Control No. 15/340,991	Applicant(s)/Patent Under Reexamination Cosic, Jasmin
	Examiner SANTIAGO GARCIA	Art Unit 2668

CLAIM		DATE								
Final	Original	06/19/2019								
	43	-								
	44	-								
	45	-								
	46	-								
	47	-								
	48	-								
	49	-								
	50	-								
	51	-								
	52	-								
	53	-								
	54	-								
	55	-								
	56	-								
	57	-								
	58	-								
	59	-								
	60	-								
	61	=								
	62	=								
	63	=								
	64	=								
	65	=								
	66	=								
	67	=								
	68	=								
	69	=								
	70	=								
	71	=								
	72	=								
	73	=								
	74	=								
	75	=								
	76	=								
	77	=								
	78	=								
	79	=								
	80	=								

Issue Classification	Application/Control No. 15/340,991	Applicant(s)/Patent Under Reexamination Cosic, Jasmin
	Examiner SANTIAGO GARCIA	Art Unit 2668

CPC						
Symbol					Type	Version
G06N	/	3	/	08	F	2013-01-01
G06N	/	5	/	022	I	2013-01-01

CPC Combination Sets								
Symbol					Type	Set	Ranking	Version
	/		/					

NONE (Assistant Examiner) (Date)	Total Claims Allowed: 20	
/SANTIAGO GARCIA/ Primary Examiner, Art Unit 2668 19 June 2019 (Primary Examiner) (Date)	O.G. Print Claim(s) 1	O.G. Print Figure 12

U.S. Patent and Trademark Office Part of Paper No.: 20190619

Issue Classification	Application/Control No. 15/340,991	Applicant(s)/Patent Under Reexamination Cosic, Jasmin
	Examiner SANTIAGO GARCIA	Art Unit 2668

INTERNATIONAL CLASSIFICATION				
CLAIMED				
G06N3/08	/	3	/	08
G06N5/02	/	5	/	02

NON-CLAIMED				
	/		/	

US ORIGINAL CLASSIFICATION	
CLASS	**SUBCLASS**

CROSS REFERENCES(S)						
CLASS	**SUBCLASS (ONE SUBCLASS PER BLOCK)**					

NONE (Assistant Examiner) (Date)	**Total Claims Allowed:** 20	
/SANTIAGO GARCIA/ 19 June 2019 Primary Examiner, Art Unit 2668 (Primary Examiner) (Date)	O.G. Print Claim(s) 1	O.G. Print Figure 12

Issue Classification	Application/Control No.	Applicant(s)/Patent Under Reexamination
	15/340,991	Cosic, Jasmin
	Examiner	**Art Unit**
	SANTIAGO GARCIA	2668

☐ Claims renumbered in the same order as presented by applicant ☐ CPA ☐ T.D. ☐ R.1.47

CLAIMS

Final	Original	Final	Original	Final	Original	Final	Original	Final	Original	Final	Original	Final	Original	Final	Original
	1		11		21		31		41		51	1	61	11	71
	2		12		22		32		42		52	2	62	12	72
	3		13		23		33		43		53	3	63	13	73
	4		14		24		34		44		54	4	64	14	74
	5		15		25		35		45		55	5	65	15	75
	6		16		26		36		46		56	6	66	16	76
	7		17		27		37		47		57	7	67	17	77
	8		18		28		38		48		58	8	68	18	78
	9		19		29		39		49		59	9	69	19	79
	10		20		30		40		50		60	10	70	20	80

NONE (Assistant Examiner)	(Date)	**Total Claims Allowed:** 20	
/SANTIAGO GARCIA/ Primary Examiner, Art Unit 2668 (Primary Examiner)	19 June 2019 (Date)	O.G. Print Claim(s) 1	O.G. Print Figure 12

U.S. Patent and Trademark Office Part of Paper No.: 20190619

Search Notes	Application/Control No.	Applicant(s)/Patent Under Reexamination
	15/340,991	Cosic, Jasmin
	Examiner	**Art Unit**
	SANTIAGO GARCIA	2668

CPC - Searched*		
Symbol	**Date**	**Examiner**
G06N3/08, G06N 5/022	06/19/2019	SG

CPC Combination Sets - Searched*		
Symbol	**Date**	**Examiner**

US Classification - Searched*			
Class	**Subclass**	**Date**	**Examiner**

* See search history printout included with this form or the SEARCH NOTES box below to determine the scope of the search.

Search Notes		
Search Notes	**Date**	**Examiner**
East and inventor name search	06/19/2019	SG
ip.com, google scholar, IEEE search same search and keywords and synonyms as east search and adopting it to each site for example ((ANTICIPAT$4 OR PREDICT$5 or determ$5) NEAR4 (INSTRUCTION NEAR4 SETS)) with many synonyms for each term along with all the different ways to search for AI	06/19/2019	SG
Consulted with Bernard Krasnic about the invention and [possible areas to search	06/19/2019	SG
Consulted with Niu, Feng with regards to 112f and how to correctly point out the claim interpretation	06/19/2019	SG
EAST search had to be re-created as some search strings were taking too long and work space ended up not working. Re-created as best as could remember.	06/19/2019	SG

/SANTIAGO GARCIA/ Primary Examiner, Art Unit 2668	

Search Notes	Application/Control No.	Applicant(s)/Patent Under Reexamination
	15/340,991	Cosic, Jasmin
	Examiner	**Art Unit**
	SANTIAGO GARCIA	2668

Interference Search

US Class/CPC Symbol	US Subclass/CPC Group	Date	Examiner
G06N3	08	06/19/2019	SG
G06N5	022	06/19/2019	SG

/SANTIAGO GARCIA/ Primary Examiner, Art Unit 2668	

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS Remove

Examiner Initial*	Cite No	Patent Number	Kind Code1	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	8996432		2015-03-31	Fu; Jicheng	
	2	6754631		2004-06-22	Din	
	3	9305216		2016-04-05	Mishra	
	4	5560011		1996-09-23	Uyama	
	5	6842877		2005-01-11	Robarts , et al.	
	6	7565340		2009-07-21	Herlocker , et al.	
	7	8261199		2012-09-04	Cradick , et al.	
	8	8266608		2012-09-11	Hecht , et al.	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	9	9268454		2016-02-23	Hamilton, II , et al.	

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS Remove

Examiner Initial*	Cite No	Publication Number	Kind Code[1]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	20120290347		2012-11-15	ELAZOUNI; ASHRAF ; et al.	
	2	20150324685		2015-11-12	Bohn; Richard Esten ; et al.	
	3	20100114746		2010-05-06	Bobbitt; Russell Patrick ; et al.	
	4	20120284026		2012-11-08	Cardillo; Peter S. ; et al.	
	5	20140211988		2014-07-31	Fan; Quanfu ; et al.	
	6	20150006171		2015-01-01	Westby; Michael C. ; et al.	
	7	20150264306		2015-09-17	Marilly; Emmanuel ; et al.	
	8	20110085734		2011-04-14	Berg; Jared S. ; et al.	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT
(Not for submission under 37 CFR 1.99)

Application Number	15340991
Filing Date	2016-11-02
First Named Inventor	Jasmin Cosic
Art Unit	
Examiner Name	
Attorney Docket Number	

	9	20150269415		2015-09-24	Gelbman; Alexander	
	10	20160274187		2016-09-22	MENON; SANKARAN M. ; et al.	
	11	20150339213		2015-11-22	LEE; Christopher Stephen ; et al.	
	12	20150310041		2015-10-29	Kier; Scott ; et al.	
	13	20160328480		2016-11-10	Owens; Erich James ; et al.	
	14	20140075249		2014-03-13	SATO; Shuhei ; et al.	
	15	20110270794		2011-11-03	Drory; Tal ; et al.	
	16	20070050719		2007-03-01	Lui; Philip ; et al.	
	17	20070061735		2007-03-15	Hoffberg; Steven M. ; et al.	
	18	20060184410		2006-08-17	Ramamurthy; Shankar ; et al.	
	19	20060265406		2006-11-23	Chkodrov; Gueorgui B. ; et al.	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT
(Not for submission under 37 CFR 1.99)

Application Number	15340991
Filing Date	2016-11-02
First Named Inventor	Jasmin Cosic
Art Unit	
Examiner Name	
Attorney Docket Number	

	20	20090287643		2009-11-19	Corville; Allen O. ; et al.	
	21	20110007079		2011-01-13	Perez et al.	
	22	20070050606		2007-03-01	Ferren et al.	
	23	20090067727		2009-03-12	HIROHATA; Hitoshi	
	24	20130007532		2013-01-03	Miller et al.	
	25	20050240412		2005-10-27	Fujita	
	26	20160167226		2016-06-16	Schnittman	
	27	20090136095		2009-05-28	Marcon et al.	
	28	20080215508		2008-09-04	Hanneman; Jeffrey E. ; et al.	
	29	20090044113		2009-02-12	Jones; Scott T. ; et al.	
	30	20130278501		2013-10-24	BULZACKI; Adrian	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99**)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number[3]	Country Code[2] i	Kind Code[4]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T[5]
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T[5]
	1	Chen et al. "Case-Based Reasoning System and Artificial Neural Networks: A Review Neural Comput & Applic (2001) 10: pp 264-276, 13 pages	
	2	JOHN J. GREFENSTETTE, CONNIE LOGGIA RAMSEY, ALAN C. SCHULTZ, Learning Sequential Decision Rules Using Simulation Models and Competition, 1990, Navy Center for Applied Research in Artificial Intelligence, Naval Research Laboratory, Washington, DC, 27 pages	
	3	ALAN C. SCHULTZ, JOHN J. GREFENSTETTE, Using a Genetic Algorithm to Learn Behaviors for Autonomous Vehicles, 1992, Navy Center for Applied Research in Artificial Intelligence, Naval Research Laboratory, Washington, DC, 12 pages	
	4	Koppula et al., "Anticipating human activities using object affordances for reactive robotic response", IEEE TRAMI 2016, published 5 May 2015, 16 pages	
	5	Orme, "System design tips for entry level smartphones - part 3", found online at "https://community.arm.com/processors/b/blog/posts/system-design-tips-for-entry-level-smartphones---part-3", Oct 21, 2013, 12 pages	
	6	NOWOSTAWSKI, MARIUSZ et al., "Dynamic Demes Parallel Genetic Algorithm", May 13, 1999, 6 pages	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number: 15340991
	Filing Date: 2016-11-02
	First Named Inventor: Jasmin Cosic
	Art Unit:
	Examiner Name:
	Attorney Docket Number:

	7	Stack Overflow, How can I quantify difference between two images?, accessed 2 August 2017 at https://stackoverflow.com/questions/189943/how-can-i-quantify-difference-between-two-images, 8 pages	
	8	SKOLICKI, ZBIGNIEW et al., "The Influence of Migration Sizes and Intervals on Island Models", June 29, 2005, 8 pages	
	9	Vahid Lari, et al., "Decentralized dynamic resource management support for massively parallel processor arrays", September 11, 2011, 8 pages	
	10	Vahid Lari, et al., "Distributed resource reservation in massively parallel processor arrays", May 16, 2011, 4 pages	
	11	Mohsen Hayati and Yazdan Shirvany, "Artificial Neural Network Approach for Short Term Load Forecasting for Illam Region", January 2007, 5 pages	
	12	Hasim Sak, Andrew Senior, and Francoise Beaufays, "Long Short-Term Memory Recurrent Neural Network Architectures for Large Scale Acoustic Modeling", January 2014, 5 pages	
	13	Jorg Walter and Klaus Schulten, "Implementation of self-organizing neural networks for visuo-motor control of an industrial robot", January 1993, 10 pages	
	14	Heikki Hyotyniemi and Aarno Lehtola, "A Universal Relation Database Interface for Knowledge Based Systems", April 1991, 5 pages	
	15	Mrissa, Michael, et al. "An avatar architecture for the web of things." IEEE Internet Computing 19.2 (2015): 30-38., 9 pages	
	16	Luck, Michael, and Ruth Aylett. "Applying artificial intelligence to virtual reality: Intelligent virtual environments." Applied Artificial Intelligence 14.1 (2000): 3-32., 30 pages	
	17	Terdjimi, Mehdi, et al. "An avatar-based adaptation workflow for the web of things." Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), 2016 IEEE 25th International Conference on. IEEE, 2016., 6 pages	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number		15340991
	Filing Date		2016-11-02
	First Named Inventor	Jasmin Cosic	
	Art Unit		
	Examiner Name		
	Attorney Docket Number		

	18		Bogdanovych, Anton, et al. "Authentic interactive reenactment of cultural heritage with 3D virtual worlds and artificial intelligence." Applied Artificial Intelligence 24.6 (2010): 617-647., 32 pages	
	19		Hernandez, Marco E. Perez, and Stephan Reiff-Marganiec. "Autonomous and self controlling smart objects for the future internet." Future internet of things and cloud (FiCloud), 2015 3rd international conference on. IEEE, 2015., 8 pages	
	20		Medini et al., "Building a Web of Things with Avatars", Managing the Web of Things (2017), 30 pages	

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature	/SANTIAGO GARCIA/	Date Considered	06/19/2019

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Jasmin Cosic/	Date (YYYY-MM-DD)	2019-05-05
Name/Print	Jasmin Cosic	Registration Number	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

EAST Search History

EAST Search History (Prior Art)

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
L1	13297	G06N3/08.cpc.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/19 14:04
L2	6613	G06N5/022.cpc.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/19 14:04
L3	19658	1 or 2	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/19 14:04
L4	1287849	AI OR (((machine near2 learn$3) (artificial adj intelligence) (decision adj tree) (random adj forrest$1) (predictive adj model$4) (association adj rule) (neural adj network$3) (bayesian adj network$3)))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/19 14:04
L5	12487	3 and L4	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/19 14:04
L6	3171	L4 AND ((ANTICIPAT$4 OR PREDICT$5 or determ$5) NEAR4 (INSTRUCTION NEAR4 SETS))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/19 14:06
L7	105	5 and L6	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/19 14:06
L8	3215	L4 AND ((ANTICIPAT$4 OR PREDICT$5	US-PGPUB;	OR	ON	2019/06/19

		or determ$5) NEAR4 (INSTRUCTION NEAR4 SETS) or (current near4 compiler))	USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB			14:19
L9	0	((G06N3/08.CPC.)).CCLS.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2019/06/19 14:25
L12	6613	G06N5/022.CPC.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/19 14:26
L13	42	12 AND 8	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/19 14:26
L14	37	((("Cosic") near3 ("Jasmin"))).INV.	US-PGPUB; USPAT; USOCR	OR	ON	2019/06/19 14:39
S1	1	"15340991"	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/17 23:31
S2	990332	AI OR (ARTIFICIAL NEAR3 INTELLIGEN$4)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/17 23:32
S3	49	S2 AND (DEVICE NEAR4 CIRCUMSTANCES WITH AUTOMAT$5)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/17 23:33
S4	123	S2 AND ((ANTICIPAT$4 OR PREDICT$5) NEAR4 (INSTRUCTION NEAR4 SETS))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/17 23:43
S5	368	S2 AND ((ANTICIPAT$4 OR PREDICT$5) WITH (INSTRUCTION NEAR4 SETS))	US-PGPUB; USPAT; USOCR;	OR	ON	2019/06/17 23:43

			FPRS; EPO; JPO; DERWENT; IBM_TDB			
S6	94	S2 AND ((ANTICIPAT$4 OR PREDICT$5) WITH (INSTRUCTION NEAR4 SETS) WITH (OBJECT OR THING OR ITEM OR DEVICE))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/17 23:43
S7	1	S2 AND ((ANTICIPAT$4 OR PREDICT$5) WITH (training near4 INSTRUCTION NEAR4 SETS) WITH (OBJECT OR THING OR ITEM OR DEVICE))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/18 00:01
S8	0	S2 AND ((ANTICIPAT$4 OR PREDICT$5) WITH (training$ near4 INSTRUCTION NEAR4 SETS with train) WITH (OBJECT OR THING OR ITEM OR DEVICE))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/18 00:01
S9	280300	("20080254429" | "4730315" | "6973446" | "7840060" | "8196119" | "7113946" | "9298749" | "20100278420" | "20080215508" | "20090287643" | "20120284026" | "7565340" | "20130238533" | "6915105" | "7240335" | "20130226974" | "8335805" | "9282309" | "9443192" | "20080288259" | "20100033780" | "5560011" | "6842877" | "20040117771" | "20070006159" | "4860203" | "6088731" | "7424705" | "7484205" | "8090669" | "8251704" | "8397227" | "8464225" | "20050289105" | "20100023541" | "20100082536" | "20160246850" | "8572035" | "20040249774" | "20050245303" | "20150055821" | "20070061735" | "20110085734" | "20150264306" | "20160328480" | "20120150773" | "20120167057" | "5602982" | "6106299" | "6314558" | "7478371" | "7721218" | "7765537" | "8019699" | "8655260" | "8667472" | "20160140999" | "20160246868" | "20030026588" | "20070106633" | "20090131152" | "20090222388" | "20130278631" | "20050240412" | "20060265406" | "20100114746" | "20130278501" | "20150006171" | "20150310041" | "20150339213" | "6754631" | "4370707" | "6126330" | "6643842" | "6728689" | "8078556" | "8195674" | "8589414" | "20030065662" | "20130218932" | "9047324" | "9367806" | "20090141969" | "20100063949" |	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/18 00:52

		"20130156345" | "20140052717" | "20090067727" | "20130007532" | "8261199" | "20080281764" | "7797259" | "7987144" | "8005828" | "8244730" | "20040194017" | "20160292185" | "7117225" | "8417740" | "20060047612" | "20090324010" | "20060184410" | "20110007079" | "20150269415" | "8996432" | "9268454" | "6735632" | "7017153" | "7249349" | "8166463" | "8364612" | "8549359" | "20050149517" | "20050149542" | "20160142650" | "20160246819" | "20070058856" | "20080144893" | "20110030031" | "20150039304" | "20090044113" | "20090136095" | "20150324685" | "20160167226" | "9305216" | "20060190930" | "20100138370" | "20110218672" | "6026234" | "6801912" | "7052277" | "7222127" | "7925984" | "8137112" | "8266608" | "8386401" | "8655900" | "20040267521" | "20090110061" | "20100241595" | "20130159021" | "20140161250" | "20140177946" | "20140207580" | "20070050606" | "20070050719" | "20110270794" | "20120290347" | "20140075249" | "20140211988" | "20160274187")".PN"				
S10	13290	G06N3/08.cpc.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/18 10:57
S11	280300	("20080254429" | "4730315" | "6973446" | "7840060" | "8196119" | "7113946" | "9298749" | "20100278420" | "20080215508" | "20090287643" | "20120284026" | "7565340" | "20130238533" | "6915105" | "7240335" | "20130226974" | "8335805" | "9282309" | "9443192" | "20080288259" | "20100033780" | "5560011" | "6842877" | "20040117771" | "20070006159" | "4860203" | "6088731" | "7424705" | "7484205" | "8090669" | "8251704" | "8397227" | "8464225" | "20050289105" | "20100023541" | "20100082536" | "20160246850" | "8572035" | "20040249774" | "20050245303" | "20150055821" | "20070061735" | "20110085734" | "20150264306" | "20160328480" | "20120150773" | "20120167057" | "5602982" | "6106299" | "6314558" | "7478371" | "7721218" | "7765537" | "8019699" | "8655260" | "8667472" | "20160140999" | "20160246868" | "20030026588" | "20070106633" |	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/18 10:58

		"20090131152" | "20090222388" | "20130278631" | "20050240412" | "20060265406" | "20100114746" | "20130278501" | "20150006171" | "20150310041" | "20150339213" | "6754631" | "4370707" | "6126330" | "6643842" | "6728689" | "8078556" | "8195674" | "8589414" | "20030065662" | "20130218932" | "9047324" | "9367806" | "20090141969" | "20100063949" | "20130156345" | "20140052717" | "20090067727" | "20130007532" | "8261199" | "20080281764" | "7797259" | "7987144" | "8005828" | "8244730" | "20040194017" | "20160292185" | "7117225" | "8417740" | "20060047612" | "20090324010" | "20060184410" | "20110007079" | "20150269415" | "8996432" | "9268454" | "6735632" | "7017153" | "7249349" | "8166463" | "8364612" | "8549359" | "20050149517" | "20050149542" | "20160142650" | "20160246819" | "20070058856" | "20080144893" | "20110030031" | "20150039304" | "20090044113" | "20090136095" | "20150324685" | "20160167226" | "9305216" | "20060190930" | "20100138370" | "20110218672" | "6026234" | "6801912" | "7052277" | "7222127" | "7925984" | "8137112" | "8266608" | "8386401" | "8655900" | "20040267521" | "20090110061" | "20100241595" | "20130159021" | "20140161250" | "20140177946" | "20140207580" | "20070050606" | "20070050719" | "20110270794" | "20120290347" | "20140075249" | "20140211988" | "20160274187")".PN"				
S12	13290	G06N3/08.cpc.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/18 11:36
S13	280300	("20080254429" | "4730315" | "6973446" | "7840060" | "8196119" | "7113946" | "9298749" | "20100278420" | "20080215508" | "20090287643" | "20120284026" | "7565340" | "20130238533" | "6915105" | "7240335" | "20130226974" | "8335805" | "9282309" | "9443192" | "20080288259" | "20100033780" | "5560011" | "6842877" | "20040117771" | "20070006159" | "4860203" | "6088731" | "7424705" | "7484205" | "8090669" | "8251704" | "8397227" | "8464225" | "20050289105" | "20100023541" |	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/18 11:36

		"20100082536" | "20160246850" | "8572035" | "20040249774" | "20050245303" | "20150055821" | "20070061735" | "20110085734" | "20150264306" | "20160328480" | "20120150773" | "20120167057" | "5602982" | "6106299" | "6314558" | "7478371" | "7721218" | "7765537" | "8019699" | "8655260" | "8667472" | "20160140999" | "20160246868" | "20030026588" | "20070106633" | "20090131152" | "20090222388" | "20130278631" | "20050240412" | "20060265406" | "20100114746" | "20130278501" | "20150006171" | "20150310041" | "20150339213" | "6754631" | "4370707" | "6126330" | "6643842" | "6728689" | "8078556" | "8195674" | "8589414" | "20030065662" | "20130218932" | "9047324" | "9367806" | "20090141969" | "20100063949" | "20130156345" | "20140052717" | "20090067727" | "20130007532" | "8261199" | "20080281764" | "7797259" | "7987144" | "8005828" | "8244730" | "20040194017" | "20160292185" | "7117225" | "8417740" | "20060047612" | "20090324010" | "20060184410" | "20110007079" | "20150269415" | "8996432" | "9268454" | "6735632" | "7017153" | "7249349" | "8166463" | "8364612" | "8549359" | "20050149517" | "20050149542" | "20160142650" | "20160246819" | "20070058856" | "20080144893" | "20110030031" | "20150039304" | "20090044113" | "20090136095" | "20150324685" | "20160167226" | "9305216" | "20060190930" | "20100138370" | "20110218672" | "6026234" | "6801912" | "7052277" | "7222127" | "7925984" | "8137112" | "8266608" | "8386401" | "8655900" | "20040267521" | "20090110061" | "20100241595" | "20130159021" | "20140161250" | "20140177946" | "20140207580" | "20070050606" | "20070050719" | "20110270794" | "20120290347" | "20140075249" | "20140211988" | "20160274187")".PN"				
S14	78	S13 and S12	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/18 11:36
S15	990516	AI OR (ARTIFICIAL NEAR3 INTELLIGEN$4)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO;	OR	ON	2019/06/18 15:18

			DERWENT; IBM_TDB			
S16	36	S15 AND ((ANTICIPAT$4 OR PREDICT$5) WITH (INSTRUCTION NEAR4 SETS)) with correlat$4	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/18 15:18
S17	100	S15 AND ((ANTICIPAT$4 OR PREDICT$5) WITH (INSTRUCTION NEAR4 SETS)) with (compar$3 or match$3 or correlat$3 or associat$3 or relat$3)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/18 15:19
S18	387	("20080254429" | "4730315" | "6973446" | "7840060" | "8196119" | "7113946" | "9298749" | "20100278420" | "20080215508" | "20090287643" | "20120284026" | "7565340" | "20130238533" | "6915105" | "7240335" | "20130226974" | "8335805" | "9282309" | "9443192" | "20080288259" | "20100033780" | "5560011" | "6842877" | "20040117771" | "20070006159" | "4860203" | "6088731" | "7424705" | "7484205" | "8090669" | "8251704" | "8397227" | "8464225" | "20050289105" | "20100023541" | "20100082536" | "20160246850" | "8572035" | "20040249774" | "20050245303" | "20150055821" | "20070061735" | "20110085734" | "20150264306" | "20160328480" | "20120150773" | "20120167057" | "5602982" | "6106299" | "6314558" | "7478371" | "7721218" | "7765537" | "8019699" | "8655260" | "8667472" | "20160140999" | "20160246868" | "20030026588" | "20070106633" | "20090131152" | "20090222388" | "20130278631" | "20050240412" | "20060265406" | "20100114746" | "20130278501" | "20150006171" | "20150310041" | "20150339213" | "6754631" | "4370707" | "6126330" | "6643842" | "6728689" | "8078556" | "8195674" | "8589414" | "20030065662" | "20130218932" | "9047324" | "9367806" | "20090141969" | "20100063949" | "20130156345" | "20140052717" | "20090067727" | "20130007532" | "8261199" | "20080281764" | "7797259" | "7987144" | "8005828" | "8244730" | "20040194017" | "20160292185" | "7117225" | "8417740" | "20060047612" | "20090324010" | "20060184410" | "20110007079" | "20150269415" |	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/18 23:12

		"8996432" | "9268454" | "6735632" | "7017153" | "7249349" | "8166463" | "8364612" | "8549359" | "20050149517" | "20050149542" | "20160142650" | "20160246819" | "20070058856" | "20080144893" | "20110030031" | "20150039304" | "20090044113" | "20090136095" | "20150324685" | "20160167226" | "9305216" | "20060190930" | "20100138370" | "20110218672" | "6026234" | "6801912" | "7052277" | "7222127" | "7925984" | "8137112" | "8266608" | "8386401" | "8655900" | "20040267521" | "20090110061" | "20100241595" | "20130159021" | "20140161250" | "20140177946" | "20140207580" | "20070050606" | "20070050719" | "20110270794" | "20120290347" | "20140075249" | "20140211988" | "20160274187").PN.				
S19	133	("20010044732" | "20020078209" | "20020161735" | "20020174189" | "20030046451" | "20030065662" | "20040010699" | "20040049476" | "20040098394" | "20040128327" | "20040203845" | "20040224675" | "20040229595" | "20040249785" | "20040249857" | "20050044165" | "20050076069" | "20050086203" | "20050149517" | "20050149542" | "20050289105" | "20060136454" | "20060212846" | "20070049246" | "20070136264" | "20070291757" | "20080005054" | "20080039062" | "20080040782" | "20080045205" | "20080086481" | "20080091763" | "20080126403" | "20080172274" | "20080189250" | "20080281825" | "20090070149" | "20090106360" | "20090176529" | "20090201908" | "20090217375" | "20090254594" | "20090275331" | "20100023531" | "20100023541" | "20100077468" | "20100082536" | "20100088316" | "20100159903" | "20100223228" | "20100257376" | "20100278162" | "20110010344" | "20110016089" | "20110106448" | "20110119758" | "20110145210" | "20110145242" | "20110161290" | "20110258345" | "20110314482" | "20120016901" | "20120101993" | "20120131116" | "20120159393" | "20120173485" | "20120183221" | "20120209925" | "20120209948" | "20130036089" | "20130198171" | "4862390" | "5121470" | "5185857" | "5533029" | "5544222" | "5598534" | "5812117" | "5956637" | "6087952" | "6122527" | "6128012" | "6202060" | "6226665" | "6493717" | "6549625" | "6636873" | "6654762" | "6795706" | "6832084" | "6871068" | "6879989" | "6889139" |	US-PGPUB; USPAT; USOCR	OR	ON	2019/06/18 23:14

		"6895234" | "6959436" | "7113946" | "7117225" | "7328030" | "7444143" | "7450962" | "7483905").PN. OR ("7523112" | "7685127" | "7756525" | "7756829" | "7779029" | "7805397" | "7813714" | "7831554" | "7865924" | "7885635" | "7945687" | "7958270" | "8031694" | "8041372" | "8099332" | "8103272" | "8145663" | "8190645" | "8204911" | "8214409" | "8266187" | "8290898" | "8291408" | "8335805" | "8359223" | "8401539" | "8412150" | "8417740").PN. OR ("9367806").URPN.				
S20	1287849	AI OR (((machine near2 learn$3) (artificial adj intelligence) (decision adj tree) (random adj forrest$1) (predictive adj model$4) (association adj rule) (neural adj network$3) (bayesian adj network$3)))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/18 23:28
S21	711	S20 AND ((ANTICIPAT$4 OR PREDICT$5) NEAR4 (INSTRUCTION NEAR4 SETS))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/18 23:29
S22	3171	S20 AND ((ANTICIPAT$4 OR PREDICT$5 or determ$5) NEAR4 (INSTRUCTION NEAR4 SETS))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/18 23:32
S23	1	S20 AND ((ANTICIPAT$4 OR PREDICT$5 or determ$5) NEAR4 (INSTRUCTION NEAR4 SETS)) with (object near4 representation)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/18 23:32
S24	126	S20 AND ((ANTICIPAT$4 OR PREDICT$5 or determ$5) NEAR4 (INSTRUCTION NEAR4 SETS)) with (represent$6)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/18 23:33
S25	152	S20 AND ((ANTICIPAT$4 OR PREDICT$5 or determ$5) NEAR4 (INSTRUCTION NEAR4 SETS)) with (represent$6 or label$4)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/18 23:33
S26	3	((ANTICIPAT$4 OR PREDICT$5 or determ$5) NEAR4 (INSTRUCTION NEAR4 SETS)) with (represent$6 or label$4) same4 (third near4 correlat$4)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/19 09:17

EAST Search History (Interference)

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
L10	0	((G06N3/08.CPC.)).CCLS.	US-PGPUB; USPAT	OR	OFF	2019/06/19 14:25
L11	0	((G06N5/022.CPC.)).CCLS.	US-PGPUB; USPAT	OR	OFF	2019/06/19 14:26

6/19/2019 2:59:01 PM
C:\Users\sgarcia\Documents\EAST\Workspaces\15340991.wsp

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

PTO/SB/08a (01-10)
Approved for use through 07/31/2012. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it contains a valid OMB control number.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS Remove

Examiner Initial*	Cite No	Patent Number	Kind Code1	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1					

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS Remove

Examiner Initial*	Cite No	Publication Number	Kind Code1	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	20030026588		2003-02-06	Elder, James H. ; et al.	
	2	20040249774		2004-12-09	Caid, William R. ; et al.	
	3	20040267521		2004-12-30	Cutler, Ross ; et al.	
	4	20050245303		2005-11-03	Graepel, Kurt Hartwig ; et al.	
	5	20060047612		2006-03-02	Stanley; Kenneth O. ; et al.	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	
Application Number	15340991
Filing Date	2016-11-02
First Named Inventor	Jasmin Cosic
Art Unit	
Examiner Name	
Attorney Docket Number	

	6	20070058856		2007-03-15	Boregowda; Lokesh R. ; et al.	
	7	20070106633		2007-05-10	Reiner; Bruce	
	8	20080144893		2008-06-19	GUO; Chun Biao ; et al.	
	9	20080288259		2008-11-20	Chambers; Robert L. ; et al.	
	10	20090110061		2009-04-30	LEE; Kang Yong ; et al.	
	11	20090131152		2009-05-21	Busse; Martin A.	
	12	20090141969		2009-06-04	Yu; Kai ; et al.	
	13	20090222388		2009-09-30	Hua; Wei ; et al.	
	14	20090324010		2009-12-31	Hou; Billy	
	15	20100033780		2010-02-11	Gitter; Kurt	
	16	20100063949		2010-03-11	EATON; JOHN ERIC ; et al.	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	17	20100241595		2010-09-23	Felsher; David Paul	
	18	20100278420		2010-11-04	Shet; Vinay Damodar ; et al.	
	19	20110030031		2011-02-03	Lussier; Paul ; et al.	
	20	20130156345		2013-06-20	Shmunk; Dmitry Valerievich	
	21	20130159021		2013-06-20	Felsher; David Paul	
	22	20130278631		2013-10-24	Border; John N. ; et al.	
	23	20140052717		2014-02-20	FRANKS; Kasian ; et al.	
	24	20140161250		2014-06-12	Leeds; Richard ; et al.	
	25	20140177946		2014-06-26	LIM; Kil-Taek ; et al.	
	26	20140207580		2014-06-24	Minnis; William Charles ; et al.	
	27	20150039304		2015-02-05	Wein; Ron	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	28	20150055821		2015-02-26	Fotland; David Allen	

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number3	Country Code2 i	Kind Code4	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T5
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T5
	1	ABOUT | OpenCV, retrieved from <URL: http://opencv.org/about.html> on Dec 13, 2014, 1 pages	
	2	Animetrics, Inc. | 3D Facial Recognition, retrieved from <URL: http://animetrics.com/> on Dec 13, 2014, 2 pages	
	3	Bag-of-words model, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	4	Beier–Neely morphing algorithm, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 1 pages	
	5	Computer vision, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 7 pages	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	6	Convolutional neural network, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 5 pages	
	7	Decimation (signal processing), retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 3 pages	
	8	Digital image processing, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 3 pages	
	9	Dissolve (filmmaking), retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 2 pages	
	10	Facial recognition system, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 6 pages	
	11	Feature detection (computer vision), retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 3 pages	
	12	Feature extraction, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	13	Gesture recognition, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 5 pages	
	14	GrabCut, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 1 pages	
	15	Image processing, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	16	Image segmentation, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 12 pages	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	17	Image warping, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 2 pages	
	18	Inbetweening, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 2 pages	
	19	Interpolation, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 5 pages	
	20	Language model, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 4 pages	
	21	List of speech recognition software, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 4 pages	
	22	Livewire Segmentation Technique, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 2 pages	
	23	Morphing, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 2 pages	
	24	Motion estimation, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	25	Muse: Face Morph || Mesh Warping, retrieved from <URL: http://alexwolfe.blogspot.com/2011/10/face-morph-mesh-warping.html> on Nov 19, 2015, 2 pages	
	26	n-gram, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 5 pages	
	27	Optical flow, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 4 pages	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	28		Outline of object recognition, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 7 pages	
	29		Phonotactics, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 3 pages	
	30		Recurrent neural network, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 5 pages	
	31		Sample rate conversion, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 2 pages	
	32		Simple interactive object extraction, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	33		Speech recognition, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 12 pages	
	34		Speech segmentation, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 4 pages	
	35		Welcome to recognize-speech.com, retrieved from <URL: http://recognize-speech.com/> on Oct 18, 2015, 1 pages	
	36		Introduction Speech, retrieved from <URL: http://recognize-speech.com/speech> on Oct 18, 2015, 1 pages	
	37		Preprocessing, retrieved from <URL: http://recognize-speech.com/preprocessing> on Oct 18, 2015, 4 pages	
	38		Feature Extraction, retrieved from <URL: http://recognize-speech.com/feature-extraction> on Oct 18, 2015, 3 pages	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	39	Acoustic model, retrieved from <URL: http://recognize-speech.com/acoustic-model> on Oct 18, 2015, 2 pages	
	40	Video content analysis, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	41	Video tracking, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 2 pages	
	42	Andrej Karpathy, Thomas Leung, George Toderici, Rahul Sukthankar, Sanketh Shetty, Li Fei-Fei, Large-scale Video Classification with Convolutional Neural Networks, Apr 14, 2014, 8 pages, Stanford University	
	43	Karen Simonyan, Andrew Zisserman, Two-Stream Convolutional Networks for Action Recognition in Videos, Nov 13, 2014, 11 pages, University of Oxford	

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature	/SANTIAGO GARCIA/	Date Considered	06/19/2019

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

 ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number		15340991
	Filing Date		2016-11-02
	First Named Inventor	Jasmin Cosic	
	Art Unit		
	Examiner Name		
	Attorney Docket Number		

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Jasmin Cosic/	Date (YYYY-MM-DD)	2016-11-10
Name/Print	Jasmin Cosic	Registration Number	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS

Remove

Examiner Initial*	Cite No	Patent Number	Kind Code1	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	7113946		2006-09-26	Cosic	
	2	7117225		2006-10-03	Cosic	
	3	8335805		2012-12-18	Cosic	
	4	8417740		2013-04-09	Cosic	
	5	8572035		2013-10-29	Cosic	
	6	8655900		2014-02-18	Cosic	
	7	9047324		2015-06-02	Cosic	
	8	9282309		2016-03-08	Cosic	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number		15340991
	Filing Date		2016-11-02
	First Named Inventor	Jasmin Cosic	
	Art Unit		
	Examiner Name		
	Attorney Docket Number		

	9	9298749		2016-03-29	Cosic	
	10	9367806		2016-06-14	Cosic	
	11	9443192		2016-09-13	Cosic	

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS Remove

Examiner Initial*	Cite No	Publication Number	Kind Code[1]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	20030065662		2003-04-03	Cosic	
	2	20040194017		2004-09-30	Cosic	
	3	20050149517		2005-07-07	Cosic	
	4	20050149542		2005-07-07	Cosic	
	5	20050289105		2005-12-29	Cosic	
	6	20100023541		2010-01-28	Cosic	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	7	20100082536		2010-04-01	Cosic	
	8	20130218932		2013-08-22	Cosic	
	9	20130226974		2013-08-29	Cosic	
	10	20160140999		2016-05-19	Cosic	
	11	20160142650		2016-05-19	Cosic	
	12	20160246819		2016-08-25	Cosic	
	13	20160246850		2016-08-25	Cosic	
	14	20160246868		2016-08-25	Cosic	
	15	20160292185		2016-10-06	Cosic	

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number3	Country Code2 i	Kind Code4	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T5

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T5
	1		

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature	/SANTIAGO GARCIA/	Date Considered	06/19/2019

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number		15340991
	Filing Date		2016-11-02
	First Named Inventor	Jasmin Cosic	
	Art Unit		
	Examiner Name		
	Attorney Docket Number		

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Jasmin Cosic/	Date (YYYY-MM-DD)	2016-11-10
Name/Print	Jasmin Cosic	Registration Number	

This collection of information is required by 37 CFR 1.97 and 1.98. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 1 hour to complete, including gathering, preparing and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS

Remove

Examiner Initial*	Cite No	Patent Number	Kind Code[1]	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	4370707		1983-01-25	Phillips , et al.	
	2	4730315		1988-03-08	Saito , et al.	
	3	4860203		1989-08-22	Corrigan , et al.	
	4	5602982		1997-02-11	Judd , et al.	
	5	6026234		2000-02-15	Hanson , et al.	
	6	6088731		2000-07-11	Kiraly , et al.	
	7	6106299		2000-08-22	Ackermann , et al.	
	8	6126330		2000-10-03	Knight	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	Application Number: 15340991
	Filing Date: 2016-11-02
	First Named Inventor: Jasmin Cosic
	Art Unit:
	Examiner Name:
	Attorney Docket Number:

	9	6314558		2001-11-06	Angel , et al.	
	10	6643842		2003-11-04	Angel , et al.	
	11	6728689		2004-04-27	Drissi , et al.	
	12	6735632		2004-05-11	Kiraly , et al.	
	13	6801912		2004-10-05	Moskowitz , et al.	
	14	6915105		2005-07-05	Masuda	
	15	6973446		2005-12-06	Mamitsuka , et al.	
	16	7017153		2006-03-21	Gouriou , et al.	
	17	7052277		2006-05-30	Kellman	
	18	7222127		2007-05-22	Bem , et al.	
	19	7240335		2007-07-03	Angel , et al.	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	
Application Number	15340991
Filing Date	2016-11-02
First Named Inventor	Jasmin Cosic
Art Unit	
Examiner Name	
Attorney Docket Number	

	20	7249349		2007-07-24	Hundt , et al.	
	21	7424705		2008-09-09	Lewis , et al.	
	22	7478371		2009-01-13	Gove	
	23	7484205		2009-01-27	Venkatapathy	
	24	7721218		2010-05-18	Awe , et al.	
	25	7765537		2010-07-27	Havin , et al.	
	26	7797259		2010-09-14	Jiang , et al.	
	27	7840060		2010-11-23	Podilchuk	
	28	7925984		2011-04-12	Awe , et al.	
	29	7987144		2011-07-26	Drissi , et al.	
	30	8005828		2011-08-23	Buchner , et al.	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	31	8019699		2011-09-13	Baxter	
	32	8078556		2011-12-13	Adi , et al.	
	33	8090669		2012-01-03	Shahani , et al.	
	34	8137112		2012-03-20	Woolf , et al.	
	35	8166463		2012-04-24	Gill , et al.	
	36	8195674		2012-06-05	Bem , et al.	
	37	8196119		2012-06-05	Gill , et al.	
	38	8244730		2012-08-14	Gupta	
	39	8251704		2012-08-28	Woolf , et al.	
	40	8266608		2012-09-11	Hecht , et al.	
	41	8364612		2013-01-29	Van Gael , et al.	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	
Application Number	15340991
Filing Date	2016-11-02
First Named Inventor	Jasmin Cosic
Art Unit	
Examiner Name	
Attorney Docket Number	

	42	8386401		2013-02-26	Virkar , et al.	
	43	8397227		2013-03-12	Fan , et al.	
	44	8464225		2013-06-11	Greifeneder	
	45	8549359		2013-10-01	Zheng	
	46	8589414		2013-11-19	Waite , et al.	
	47	8655260		2014-02-18	Yueh , et al.	
	48	8667472		2014-03-04	Molinari	

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS Remove

Examiner Initial*	Cite No	Publication Number	Kind Code1	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	20120167057		2012-06-28	Schmich; Christopher P. ; et al.	
	2	20120150773		2012-06-14	DiCorpo; Phillip ; et al.	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	3	20080254429		2008-10-16	Woolf; Susan D. ; et al.	
	4	20040117771		2004-06-17	Venkatapathy, Ramanathan	
	5	20130238533		2013-09-12	VIRKAR; Hemant ; et al.	
	6	20060190930		2006-08-24	Hecht; Daniel M. ; et al.	
	7	20070006159		2007-01-04	Hecht; Daniel Michael ; et al.	
	8	20080281764		2008-11-13	Baxter; Jonathan	
	9	20100138370		2010-06-03	Wu; Wang ; et al.	
	10	20110218672		2011-09-08	Maisonnier; Bruno ; et al.	

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number3	Country Code2 i	Kind Code4	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T5
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T5
	1	About Event Tracing, retrieved from <URL: http://msdn.microsoft.com/en-us/library/aa363668(d=default,l=en-us,v=vs.85).aspx> on Jan 12, 2014, 2 pages	
	2	Add dynamic Java code to your application, retrieved from <URL: http://www.javaworld.com/article/2071777/design-patterns/add-dynamic-java-code-to-your-application.html> on May 13, 2014, 6 pages	
	3	BCEL, retrieved from <URL: http://commons.apache.org> on May 13, 2014, 2 pages	
	4	Artificial intelligence, retrieved from <URL: http://wikipedia.com> on May 21, 2014, 28 pages	
	5	Artificial neural network, retrieved from <URL: http://wikipedia.com> on May 21, 2014, 12 pages	
	6	Branch (computer science), retrieved from <URL: http://wikipedia.com> on May 6, 2014, 2 pages	
	7	Branch trace, retrieved from <URL: http://wikipedia.com> on May 28, 2014, 2 pages	
	8	C Function Call Conventions and the Stack, retrieved from <URL: http://www.csee.umbc.edu/~chang/cs313.s02/stack.shtml> on Jan 11, 2014, 4 pages	
	9	Call stack, retrieved from <URL: http://wikipedia.com> on Jan 11, 2014, 5 pages	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	10	Program compilation and execution flow, retrieved from <URL: http://cs.stackexchange.com/questions/6187/program-compilation-and-execution-flow> on Jan 9, 2014, 2 pages	
	11	Control unit, retrieved from <URL: http://wikipedia.com> on May 25, 2014, 2 pages	
	12	Create dynamic applications with javax.tools, retrieved from <URL: http://www.ibm.com/developerworks/library/j-jcomp/> on May 13, 2014, 11 pages	
	13	Creating a Debugging and Profiling Agent with JVMTI, retrieved from <URL: http://www.oracle.com/technetwork/articles/java/jvmti-136367.html> on Jan 11, 2014, 18 pages	
	14	Decision tree learning, retrieved from <URL: http://wikipedia.com> on Jun 24, 2014, 5 pages	
	15	Dynamic Bytecode Instrumentation, retrieved from <URL: http://www.drdobbs.com/tools/dynamic-bytecode-instrumentation/184406433> on Apr 26, 2014, 6 pages	
	16	Dynamic programming language, retrieved from <URL: http://wikipedia.com> on May 7, 2014, 4 pages	
	17	Dynamic recompilation, retrieved from <URL: http://wikipedia.com> on May 19, 2014, 4 pages	
	18	eval(), retrieved from <URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/eval> on May 7, 2014, 13 pages	
	19	Function, retrieved from <URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Function> on May 7, 2014, 11 pages	
	20	FunctionEnter2 Function, retrieved from <URL: https://msdn.microsoft.com/en-us/library/aa964981(d=default,l=en-us,v=vs.110).aspx> on Mar 19, 2014, 2 pages	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	
Application Number	15340991
Filing Date	2016-11-02
First Named Inventor	Jasmin Cosic
Art Unit	
Examiner Name	
Attorney Docket Number	

	21	FunctionLeave2 Function, retrieved from <URL: https://msdn.microsoft.com/en-us/library/aa964942(d=default,l=en-us,v=vs.110).aspx> on Mar 19, 2014, 1 pages	
	22	Heuristic, retrieved from <URL: http://wikipedia.com> on May 22, 2014, 5 pages	
	23	How to: Configure Trace Switches, retrieved from <URL: https://msdn.microsoft.com/en-us/library/vstudio/t06xyy08(d=default,l=en-us,v=vs.100).aspx> on Mar 19, 2014, 3 pages	
	24	Indirect branch, retrieved from <URL: http://wikipedia.com> on May 5, 2014, 1 pages	
	25	Instruction cycle, retrieved from <URL: http://wikipedia.com> on Apr 30, 2014, 2 pages	
	26	Instruction set, retrieved from <URL: http://wikipedia.com> on Apr 29, 2014, 6 pages	
	27	Interrupt, retrieved from <URL: http://wikipedia.com> on May 15, 2014, 5 pages	
	28	Introduction to Instrumentation and Tracing, retrieved from <URL: http://msdn.microsoft.com/en-us/library/aa983649(d=default,l=en-us,v=vs.71).aspx> on Jan 10, 2014, 2 pages	
	29	Java bytecode, retrieved from <URL: http://wikipedia.com> on May 27, 2014, 4 pages	
	30	Package java.util.logging, retrieved from <URL: http://docs.oracle.com/javase/1.5.0/docs/api/java/util/logging/package-summary.html> on Mar 19, 2014, 2 pages	
	31	javac - Java programming language compiler, retrieved from <URL: http://docs.oracle.com/javase/7/docs/technotes/tools/windows/javac.html> on May 13, 2014, 12 pages	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number: 15340991
	Filing Date: 2016-11-02
	First Named Inventor: Jasmin Cosic
	Art Unit:
	Examiner Name:
	Attorney Docket Number:

	32		Interface JavaCompiler, retrieved from <URL: http://wikipedia.com> on May 13, 2014, 4 pages	
	33		Javassist, retrieved from <URL: http://wikipedia.com> on May 13, 2014, 2 pages	
	34		Just-in-time compilation, retrieved from <URL: http://wikipedia.com> on May 19, 2014, 5 pages	
	35		JVM Tool Interface, retrieved from <URL: http://docs.oracle.com/javase/7/docs/platform/jvmti/jvmti.html> on Apr 26, 2014, 127 pages	
	36		Machine code, retrieved from <URL: http://wikipedia.com> on Apr 30, 2014, 3 pages	
	37		Metaprogramming, retrieved from <URL: http://wikipedia.com> on May 8, 2014, 3 pages	
	38		Microsoft.VisualBasic.Logging Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/ms128122(d=default,l=en-us,v=vs.110).aspx> on Jan 13, 2014, 1 pages	
	39		Pin (computer program), retrieved from <URL: http://wikipedia.com> on May 6, 2014, 3 pages	
	40		Processor register, retrieved from <URL: http://wikipedia.com> on Apr 29, 2014, 4 pages	
	41		Profiling (computer programming), retrieved from <URL: http://wikipedia.com> on Jan 10, 2014, 4 pages	
	42		Profiling Overview, retrieved from <URL: http://msdn.microsoft.com/en-us/library/bb384493(d=default,l=en-us,v=vs.110).aspx> on Jan 12, 2014, 6 pages	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	43	Program counter, retrieved from <URL: http://wikipedia.com> on Apr 24, 2014, 2 pages	
	44	ptrace, retrieved from <URL: http://wikipedia.com> on Apr 24, 2014, 2 pages	
	45	Reflection (computer programming), retrieved from <URL: http://wikipedia.com> on May 7, 2014, 5 pages	
	46	Class Runtime, retrieved from <URL: http://docs.oracle.com/javase/7/docs/api/java/lang/Runtime.html> on Jan 10, 2014, 10 pages	
	47	Self-modifying code, retrieved from <URL: http://wikipedia.com> on Jan 12, 2014, 6 pages	
	48	Subjective logic, retrieved from <URL: http://wikipedia.com> on May 22, 2014, 5 pages	
	49	System.CodeDom.Compiler Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/z6b99ydt(d=default,l=en-us,v=vs.110).aspx> on May 13, 2014, 2 pages	
	50	System.Reflection.Emit Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/xd5fw18y(d=default,l=en-us,v=vs.110).aspx> on May 13, 2014, 2 pages	

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature	/SANTIAGO GARCIA/	Date Considered	06/19/2019

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Jasmin Cosic/	Date (YYYY-MM-DD)	2016-11-10
Name/Print	Jasmin Cosic	Registration Number	

 ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS Remove

Examiner Initial*	Cite No	Patent Number	Kind Code1	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1					

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS Remove

Examiner Initial*	Cite No	Publication Number	Kind Code1	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1					

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number3	Country Code2 i	Kind Code4	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T5
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T5

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	1	Tracing (software), retrieved from <URL: http://wikipedia.com> on Jan 10, 2014, 3 pages	
	2	Tree (data structure), retrieved from <URL: http://wikipedia.com> on Jun 24, 2014, 6 pages	
	3	PTRACE(2), retrieved from <URL: http://unixhelp.ed.ac.uk/CGI/man-cgi?ptrace> on Mar 19, 2014, 5 pages	
	4	Wevtutil, retrieved from <URL: http://technet.microsoft.com/en-us/library/cc732848(d=default,l=en-us,v=ws.11).aspx> on Apr 28, 2014, 5 pages	
	5	Intel Processor Trace, retrieved from <URL: https://software.intel.com/en-us/blogs/2013/09/18/processor-tracing> on Apr 28, 2014, 3 pages	
	6	YOUNGHOON JUNG, JAVA DYNAMICS Reflection and a lot more, Oct 10, 2012, 55 pages, Columbia University	
	7	AMITABH SRIVASTAVA, ALAN EUSTACE, ATOM A System for Building Customized Program Analysis Tools, May 3, 2004, 12 pages	
	8	MATHEW SMITHSON, KAPIL ANAND, APARNA KOTHA, KHALED ELWAZEER, NATHAN GILES, RAJEEV BARUA, Binary Rewriting without Relocation Information, Nov 10, 2010, 11 pages, University of Maryland	
	9	MAREK OLSZEWSKI, KEIR MIERTE, ADAM CZAJKOWSKI, ANGELA DEMLE BROWN, JIT Instrumentation - A Novel Approach To Dynamically Instrument Operating Systems, Feb 12, 2007, 14 pages, University of Toronto	

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature	/SANTIAGO GARCIA/	Date Considered	06/19/2019

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Jasmin Cosic/	Date (YYYY-MM-DD)	2016-11-10
Name/Print	Jasmin Cosic	Registration Number	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Inventor:	Jasmin Cosic	Confirmation No.: 1993
Title:	"ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION"	
Serial No.:	15/340,991	Filed: November 2, 2016
Examiner:	GARCIA, SANTIAGO	Group Art Unit: 2668

Via EFS-Web July 18, 2019

COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, VA 22313-1450

COMMENTS ON STATEMENT OF REASONS FOR ALLOWANCE

Dear Commissioner:

In response to the Notice of Allowance (hereinafter "Notice of Allowance") dated June 24, 2019 concerning the above-identified application, the applicant respectfully submits the following Comments on Statement of Reasons for Allowance for the record.

A good summary of the claimed invention can be found in the abstract and independent claim 61.

With respect to at least some of the elements of the claimed invention that

prior art does not disclose, the Examiner notes that the claimed invention is allowable at least because “the prior art does not disclose the device that operates as the user directs it, hence, does not disclose learning the device operation from the user directing the device. Prior art discloses a system that explores its own environment on its own, which is very different from relying on the user to direct it. Prior art does not disclose the first and the second correlations that each include a circumstance representation correlated with instruction sets. Prior art does not disclose the knowledgebase, which includes a 3-level nested data structure recited in the independent claims. Such as individual circumstance representation (a data structure itself) and individual instruction set (a data structure itself when learned in the learning process). And correlation (a data structure itself) that includes a circumstance representation correlated with instruction sets. And knowledgebase a data structure itself including a plurality of correlation data structures in various arrangements. Further autonomous operation of the device in the claims performs the operation and does not rely on the re-training such as in the prior art as noted below, and the re-training is a crucial function in the proper operating of the systems in the prior art.” Each of the above-noted individual elements that prior art does not disclose and especially some or all of them combined render all claims of the claimed invention allowable.

With respect to at least some of the differences between the claimed invention and Schnittman, the training in Schnittman is different than the learning process recited in independent claims 61, 78, and 79 because the training in Schnittman does not include a user operating a device recited in independent claims 61, 78, and 79, and what is learned in the training in Schnittman is different than what is learned (i.e. at least a portion of a first and a second correlations each correlation including one or more circumstance representations correlated with one or more instruction sets, etc.) in the learning process recited in independent claims 61, 78, and 79. Therefore, the learning process itself is

different and the artificial knowledge learned in the learning process is different. Further, robot behavior or determining robot behavior in Schnittman is not the same as the "first one or more instruction sets for operating the first device learned in the learning process" recited in independent claims 61, 78, and 79 since robot behavior in Schnittman is not learned in a learning process that includes a user operating the robot. In fact, in Fig. 4 "Programmed Behaviors 430", Schnittman teaches programmed robot behavior and, therefore, teaches away from instruction sets learned in a learning process that includes a user operating a device recited in independent claims 61, 78, and 79. Further, images, buffer, and database in Schnittman 545 are all different than a correlation including one or more circumstance representations correlated with one or more instruction sets. Further, the classifier in Schnittman is not the same as any element of independent claims 61, 78, and 79.

The above-noted include only a few of many claimed limitations that prior art does not disclose and other differences. In view of at least the above-noted claimed limitations that prior art does not disclose and other differences, the prior art references do not disclose the claimed invention, the specifics of the claimed invention, nor the broad spectrum idea.

The applicant respectfully submits that the allowed claims should not be construed as an admission that the claims as previously presented are not patentable, or that the cited references anticipate or render obvious the claims as previously presented. Indeed, the applicant respectfully submits that the claims as previously presented, and/or further amendments thereto, also contain patentable subject material. Thus, the applicant reserves the right to pursue the claims as previously presented, and/or further amendments thereto, in one or more continuation applications.

If the Examiner would like to discuss any aspect of this application, the

Inventor: Jasmin Cosic
Serial No.: 15/340,991
Filing Date: November 2, 2016

Examiner is respectfully requested to contact the undersigned at (317) 772-1312.

I hereby certify that this correspondence is being submitted electronically via EFS Web to the United States Patent and Trademark Office.

By /Jasmin Cosic/
Jasmin Cosic

Date submitted: July 18, 2019

Respectfully submitted,

/Jasmin Cosic/

Jasmin Cosic

Electronic Acknowledgement Receipt

EFS ID:	36629789
Application Number:	15340991
International Application Number:	
Confirmation Number:	1993
Title of Invention:	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION
First Named Inventor/Applicant Name:	Jasmin Cosic
Customer Number:	116094
Filer:	Jasmin Cosic
Filer Authorized By:	
Attorney Docket Number:	
Receipt Date:	18-JUL-2019
Filing Date:	02-NOV-2016
Time Stamp:	18:24:33
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	no

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1	Miscellaneous Incoming Letter	COMMENTS_ON_STATEMENT_OF_REASONS_FOR_ALLOWANCE.pdf	23561 7e822656e6102617ecc7c66e547b32ce057768ea	no	4

Warnings:

Information:	
Total Files Size (in bytes):	23561

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.
National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.
New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Inventor:	Jasmin Cosic	Confirmation No.: 1993
Title:	"ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION"	
Serial No.:	15/340,991	Filed: November 2, 2016
Examiner:	GARCIA, SANTIAGO	Group Art Unit: 2668

Via EFS-Web July 27, 2019

Mail Stop Amendment
COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, VA 22313-1450

AMENDMENT PURSUANT TO 37 C.F.R. 1.312

Dear Commissioner:

In response to notice of allowance (hereinafter "Notice of Allowance") dated June 24, 2019 and pursuant to 37 C.F.R. 1.312, the applicant hereby amends the listing of claims that the Examiner filed in the Examiner's amendment in the Notice of Allowance.

Listing of Claims

61. (currently amended) A method implemented using a computing system that includes one or more processor circuits, the method comprising:

accessing a memory that stores at least a knowledgebase that includes: a first correlation including a first circumstance representation correlated with a first one or more instruction sets for operating a first device and a second correlation including a second circumstance representation correlated with a second one or more instruction sets for operating the first device, wherein the first circumstance representation represents a first circumstance detected at least in part by one or more sensors of the first device and the second circumstance representation represents a second circumstance detected at least in part by the one or more sensors of the first device, and wherein at least a portion of the first correlation and at least a portion of the second correlation are learned in a learning process that includes operating the first device at least partially by a user;

generating or receiving a third circumstance representation, wherein the third circumstance representation represents a third circumstance detected at least in part by the one or more sensors of the first device or at least in part by one or more sensors of a second device;

anticipating the first one or more instruction sets for operating the first device learned in the learning process based on at least partial match between the third circumstance representation and the first circumstance representation; and

at least in response to the anticipating, executing the first one or more instruction sets for operating the first device learned in the learning process, wherein the first device or ~~[[a]]~~ the second device autonomously performs one or more operations defined by the first one or more instruction sets for operating the first device learned in the learning process.

62. (previously presented) The method of claim 61, wherein the knowledgebase further includes: a first knowledge cell and a second knowledge cell, and wherein the first knowledge cell includes the first correlation and the second knowledge cell includes the second correlation.

63. (previously presented) The method of claim 61, wherein the learning process includes: creating, inserting, deleting, modifying, or manipulating an element of the first correlation, or creating, inserting, deleting, modifying, or manipulating an element of the second correlation.

64. (previously presented) The method of claim 61, wherein the learning process includes:

generating or receiving the first circumstance representation, and generating or receiving the second circumstance representation; and

obtaining or receiving the first one or more instruction sets for operating the first device, and obtaining or receiving the second one or more instruction sets for operating the first device, and wherein the knowledgebase further

includes a fourth correlation including a fourth circumstance representation correlated with a fourth one or more instruction sets for operating the second device, and wherein the fourth circumstance representation represents a fourth circumstance detected at least in part by the one or more sensors of the second device, and wherein at least a portion of the fourth correlation is learned in another learning process that includes:

generating or receiving the fourth circumstance representation; and

obtaining or receiving the fourth one or more instruction sets for operating the second device.

65. (currently amended) The method of claim 61, wherein the anticipating the first one or more instruction sets for operating the first device learned in the learning process based on the at least partial match between the third circumstance representation and the first circumstance representation includes:

determining that a number of at least partially matching portions of the third circumstance representation and portions of the first circumstance representation exceeds a threshold number[[;]], or

determining that a percentage of at least partially matching portions of the third circumstance representation and portions of the first circumstance representation exceeds a threshold percentage.

66. (previously presented) The method of claim 61, wherein the at least the portion of the first correlation and the at least the portion of the second

correlation are learned in the learning process while the user operates the first device, and wherein the one or more operations defined by the first one or more instruction sets for operating the first device learned in the learning process correspond to the user's methodology of operating the first device learned in the learning process.

67. (previously presented) The method of claim 61, wherein the knowledgebase further includes a fourth correlation including a fourth circumstance representation correlated with a fourth one or more instruction sets for operating: the first device, the second device, or a third device, and wherein a first connection is generated to connect the first correlation with the second correlation, and wherein a second connection is generated to connect the second correlation with the fourth correlation.

68. (previously presented) The method of claim 61, further comprising:

modifying the first one or more instruction sets for operating the first device learned in the learning process or a copy of the first one or more instruction sets for operating the first device learned in the learning process, and wherein the executing the first one or more instruction sets for operating the first device learned in the learning process includes executing the modified the first one or more instruction sets for operating the first device learned in the learning process or the modified the copy of the first one or more instruction sets for operating the first device learned in the learning process, and wherein the

performing, by the first device or by the second device, the one or more operations defined by the first one or more instruction sets for operating the first device learned in the learning process includes performing, by the first device or by the second device, one or more operations defined by the modified the first one or more instruction sets for operating the first device learned in the learning process or by the modified the copy of the first one or more instruction sets for operating the first device learned in the learning process.

69. (previously presented) The method of claim 61, wherein the knowledgebase further includes a fourth correlation including a fourth circumstance representation correlated with a fourth one or more instruction sets for operating the first device, and wherein the fourth circumstance representation represents a fourth circumstance detected at least in part by the one or more sensors of the first device, and wherein at least a portion of the fourth correlation is learned in another learning process that includes operating the first device at least partially by the user.

70. (previously presented) The method of claim 61, wherein the knowledgebase further includes a fourth correlation including a fourth circumstance representation correlated with a fourth one or more instruction sets for operating the first device, and wherein the fourth circumstance representation represents a fourth circumstance detected at least in part by the one or more sensors of the first device, and wherein at least a portion of the fourth correlation is learned in

another learning process that includes operating the first device at least partially by another user.

71. (previously presented) The method of claim 61, wherein the knowledgebase further includes a fourth correlation including a fourth circumstance representation correlated with a fourth one or more instruction sets for operating the second device, and wherein the fourth circumstance representation represents a fourth circumstance detected at least in part by the one or more sensors of the second device, and wherein at least a portion of the fourth correlation is learned in another learning process that includes operating the second device at least partially by another user.

72. (previously presented) The method of claim 61, wherein the first circumstance includes one or more objects detected at least in part by the one or more sensors of the first device at a first time or during a first time period, and wherein the second circumstance includes one or more objects detected at least in part by the one or more sensors of the first device at a second time or during a second time period, and wherein the third circumstance includes: one or more objects detected at least in part by the one or more sensors of the first device at a third time or during a third time period, or one or more objects detected at least in part by the one or more sensors of the second device at a third time or during a third time period.

73. (currently amended) The method of claim 61, wherein the first circumstance representation is a data structure including one or more data about the first circumstance of the first device, and wherein the second circumstance representation is a data structure including one or more data about the second circumstance of the first device, and wherein the third circumstance representation is a data structure including one or more data about: the third circumstance of the first device, or the third circumstance of the second device.

74. (previously presented) The method of claim 61, wherein the first circumstance representation includes: one or more object representations, or one or more collections of object representations, and wherein the second circumstance representation includes: one or more object representations, or one or more collections of object representations, and wherein the third circumstance representation includes: one or more object representations, or one or more collections of object representations.

75. (currently amended) The method of claim 61, wherein, to correlate the first circumstance representation with the first one or more instruction sets for operating the first device, a determination is made that the first one or more instruction sets for operating the first device temporally correspond to the first circumstance representation, and wherein the first circumstance representation includes: one or more object representations, or one or more collections of object representations.

76. (previously presented) The method of claim 61, wherein the anticipating the first one or more instruction sets for operating the first device learned in the learning process based on the at least partial match between the third circumstance representation and the first circumstance representation includes determining the first one or more instruction sets for operating the first device learned in the learning process based on the at least partial match between the third circumstance representation and the first circumstance representation.

77. (previously presented) The method of claim 61, wherein elements of the computing system are included in: a single device, or multiple devices, and wherein the one or more processor circuits include: one or more microcontrollers, one or more computing circuits, or one or more electronic circuits, and wherein the memory includes: a volatile memory, or a non-volatile memory, and wherein the first device includes: a robot, a vehicle, an appliance, an electronic device, or a mechanical machine, and wherein the second device includes: a robot, a vehicle, an appliance, an electronic device, or a mechanical machine, and wherein an instruction set of the first one or more instruction sets for operating the first device includes at least one selected from the group comprising: only one instruction, a plurality of instructions, one or more inputs, one or more commands, one or more computer commands, one or more keywords, one or more symbols, one or more operators, one or more variables, one or more values, one or more objects or object references, one or more data structures or

data structure references, one or more functions or function references, one or more parameters, one or more signals, one or more characters, one or more digits, one or more user operating directions, one or more user directions, one or more user inputs, one or more representations of one or more user actions, one or more representations of one or more user clicks, one or more binary bits, one or more assembly language commands, one or more states, one or more codes, one or more data, and one or more information, and wherein an instruction set of the second one or more instruction sets for operating the first device includes at least one selected from the group comprising: only one instruction, a plurality of instructions, one or more inputs, one or more commands, one or more computer commands, one or more keywords, one or more symbols, one or more operators, one or more variables, one or more values, one or more objects or object references, one or more data structures or data structure references, one or more functions or function references, one or more parameters, one or more signals, one or more characters, one or more digits, one or more user operating directions, one or more user directions, one or more user inputs, one or more representations of one or more user actions, one or more representations of one or more user clicks, one or more binary bits, one or more assembly language commands, one or more states, one or more codes, one or more data, and one or more information, and wherein the one or more sensors of the first device include at least one selected from the group comprising: one or more cameras, one or more microphones, one or more radars, one or more lidars, one or more sonars, and one or more apparatuses for detecting objects or object properties,

and wherein the one or more sensors of the second device include at least one selected from the group comprising: one or more cameras, one or more microphones, one or more radars, one or more lidars, one or more sonars, and one or more apparatuses for detecting objects or object properties, and wherein the at least the portion of the first correlation includes: one portion of the first correlation, multiple portions of the first correlation, all portions of the first correlation, or the entire first correlation, and wherein the at least the portion of the second correlation includes: one portion of the second correlation, multiple portions of the second correlation, all portions of the second correlation, or the entire second correlation, and wherein an object of the first circumstance is the same as an object of the third circumstance, or multiple objects of the first circumstance are the same as multiple objects of the third circumstance, or all objects of the first circumstance are the same as all objects of the third circumstance, or all objects of the first circumstance are different than all objects of the third circumstance.

78. (currently amended) One or more non-transitory machine readable media storing machine readable code that when executed by one or more processor circuits causes the one or more processor circuits to perform at least:

accessing a memory that stores at least a knowledgebase that includes: a first correlation including a first circumstance representation correlated with a first one or more instruction sets for operating a first device and a second correlation including a second circumstance representation correlated with a second one or

more instruction sets for operating the first device, wherein the first circumstance representation represents a first circumstance detected at least in part by one or more sensors of the first device and the second circumstance representation represents a second circumstance detected at least in part by the one or more sensors of the first device, and wherein at least a portion of the first correlation and at least a portion of the second correlation are learned in a learning process that includes operating the first device at least partially by a user;

generating or receiving a third circumstance representation, wherein the third circumstance representation represents a third circumstance detected at least in part by the one or more sensors of the first device or at least in part by one or more sensors of a second device;

anticipating the first one or more instruction sets for operating the first device learned in the learning process based on at least partial match between the third circumstance representation and the first circumstance representation; and

at least in response to the anticipating, causing the first device or ~~[[a]]~~ the second device to perform one or more operations defined by the first one or more instruction sets for operating the first device learned in the learning process at least by causing the one or more processor circuits or another one or more processor circuits to execute the first one or more instruction sets for operating the first device learned in the learning process.

79. (currently amended) A system comprising:

a memory that stores at least a knowledgebase that includes: a first correlation including a first circumstance representation correlated with a first one or more instruction sets for operating a first device and a second correlation including a second circumstance representation correlated with a second one or more instruction sets for operating the first device, wherein the first circumstance representation represents a first circumstance detected at least in part by one or more sensors of the first device and the second circumstance representation represents a second circumstance detected at least in part by the one or more sensors of the first device, and wherein at least a portion of the first correlation and at least a portion of the second correlation are learned in a learning process that includes operating the first device at least partially by a user;

means for generating or receiving a third circumstance representation, wherein the third circumstance representation represents a third circumstance detected at least in part by the one or more sensors of the first device or at least in part by one or more sensors of a second device;

means for anticipating the first one or more instruction sets for operating the first device learned in the learning process based on at least partial match between the third circumstance representation and the first circumstance representation; and

means for executing, at least in response to the anticipating, the first one or more instruction sets for operating the first device learned in the learning process, wherein the first device or ~~[[a]]~~ the second device autonomously

performs one or more operations defined by the first one or more instruction sets for operating the first device learned in the learning process.

80. (previously presented) The system of claim 79, wherein the means for generating or receiving the third circumstance representation includes one or more processor circuits, and wherein the means for anticipating the first one or more instruction sets for operating the first device learned in the learning process based on the at least partial match between the third circumstance representation and the first circumstance representation includes one or more processor circuits, and wherein the means for executing, at least in response to the anticipating, the first one or more instruction sets for operating the first device learned in the learning process includes one or more processor circuits.

Inventor: Jasmin Cosic
Serial No.: 15/340,991
Filing Date: November 2, 2016

Remarks

I. 37 C.F.R. 1.312 Amendment

Entry of the foregoing amendments is respectfully requested pursuant to 37 C.F.R. 1.312.

The amendments to claims 61, 65, 73, 75, 78, and 79 are needed to correct minor typographical errors as indicated.

No additional search or examination is needed because the claims correct minor typographical errors. The claims are patentable for the same reasons as those outlined in the Notice of Allowance.

If the Examiner would like to discuss any aspect of this application, the Examiner is respectfully requested to contact the undersigned at (317) 772-1312.

I hereby certify that this correspondence is being submitted electronically via EFS Web to the United States Patent and Trademark Office.

By /Jasmin Cosic/
Jasmin Cosic

Date submitted: July 27, 2019

Respectfully submitted,

/Jasmin Cosic/

Jasmin Cosic

Electronic Acknowledgement Receipt

EFS ID:	36707654
Application Number:	15340991
International Application Number:	
Confirmation Number:	1993
Title of Invention:	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION
First Named Inventor/Applicant Name:	Jasmin Cosic
Customer Number:	116094
Filer:	Jasmin Cosic
Filer Authorized By:	
Attorney Docket Number:	
Receipt Date:	27-JUL-2019
Filing Date:	02-NOV-2016
Time Stamp:	21:34:14
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	no

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1		312_Amendment.pdf	57078 a154a65003ae6c42ae7b7d16122246d6ccd2ff4b	yes	15

Multipart Description/PDF files in .zip description		
Document Description	**Start**	**End**
Amendment after Notice of Allowance (Rule 312)	1	1
Claims	2	14
Applicant Arguments/Remarks Made in an Amendment	15	15

Warnings:

Information:

Total Files Size (in bytes):	57078

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

<u>New Applications Under 35 U.S.C. 111</u>
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.
<u>National Stage of an International Application under 35 U.S.C. 371</u>
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.
<u>New International Application Filed with the USPTO as a Receiving Office</u>
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
15/340,991	11/02/2016	Jasmin Cosic		1993

116094 7590 08/08/2019
Jasmin Cosic
108 Woodbury Street
Pawtucket, RI 02861

EXAMINER	
GARCIA, SANTIAGO	
ART UNIT	**PAPER NUMBER**
2668	
MAIL DATE	**DELIVERY MODE**
08/08/2019	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Response to Rule 312 Communication	**Application No.** 15/340,991	**Applicant(s)** Cosic, Jasmin	
	Examiner SANTIAGO GARCIA	**Art Unit** 2668	**AIA Status** Yes

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

1. ☑ The amendment filed on 27 July 2019 under 37 CFR 1.312 has been considered, and has been:

a) ☐ entered.

b) ☑ entered as directed to matters of form not affecting the scope of the invention.

c) ☐ disapproved because the amendment was filed after the payment of the issue fee.

Any amendment filed after the date the issue fee is paid must be accompanied by a petition under 37 CFR 1.313(c)(1) and the required fee to withdraw the application from issue.

d) ☐ disapproved. See explanation below.

e) ☐ entered in part. See explanation below.

/SANTIAGO GARCIA/
Primary Examiner, Art Unit 2668

PART B - FEE(S) TRANSMITTAL

Complete and send this form, together with applicable fee(s), by mail or fax, or via EFS-Web.

By mail, send to: Mail Stop ISSUE FEE
Commissioner for Patents
P.O. Box 1450
Alexandria, Virginia 22313-1450

By fax, send to: (571)-273-2885

INSTRUCTIONS: This form should be used for transmitting the ISSUE FEE and PUBLICATION FEE (if required). Blocks 1 through 5 should be completed where appropriate. All further correspondence including the Patent, advance orders and notification of maintenance fees will be mailed to the current correspondence address as indicated unless corrected below or directed otherwise in Block 1, by (a) specifying a new correspondence address; and/or (b) indicating a separate "FEE ADDRESS" for maintenance fee notifications.

CURRENT CORRESPONDENCE ADDRESS (Note: Use Block 1 for any change of address)

116094 7590 06/24/2019

Jasmin Cosic
108 Woodbury Street
Pawtucket, RI 02861

Note: A certificate of mailing can only be used for domestic mailings of the Fee(s) Transmittal. This certificate cannot be used for any other accompanying papers. Each additional paper, such as an assignment or formal drawing, must have its own certificate of mailing or transmission.

Certificate of Mailing or Transmission

I hereby certify that this Fee(s) Transmittal is being deposited with the United States Postal Service with sufficient postage for first class mail in an envelope addressed to the Mail Stop ISSUE FEE address above, or being transmitted to the USPTO via EFS-Web or by facsimile to (571) 273-2885, on the date below.

Jasmin Cosic (Typed or printed name)
/Jasmin Cosic/ (Signature)
August 16, 2019 (Date)

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
15/340,991	11/02/2016	Jasmin Cosic		1993

TITLE OF INVENTION: ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION

APPLN. TYPE	ENTITY STATUS	ISSUE FEE DUE	PUBLICATION FEE DUE	PREV. PAID ISSUE FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	SMALL	$500	$0.00	$0.00	$500	09/24/2019

EXAMINER	ART UNIT	CLASS-SUBCLASS
GARCIA, SANTIAGO	2668	382-155000

1. Change of correspondence address or indication of "Fee Address" (37 CFR 1.363).

☐ Change of correspondence address (or Change of Correspondence Address form PTO/SB/122) attached.

☐ "Fee Address" indication (or "Fee Address" Indication form PTO/SB/47; Rev 03-09 or more recent) attached. **Use of a Customer Number is required.**

2. For printing on the patent front page, list
(1) The names of up to 3 registered patent attorneys or agents OR, alternatively,
(2) The name of a single firm (having as a member a registered attorney or agent) and the names of up to 2 registered patent attorneys or agents. If no name is listed, no name will be printed.

1 ______
2 ______
3 ______

3. ASSIGNEE NAME AND RESIDENCE DATA TO BE PRINTED ON THE PATENT (print or type)

PLEASE NOTE: Unless an assignee is identified below, no assignee data will appear on the patent. If an assignee is identified below, the document must have been previously recorded, or filed for recordation, as set forth in 37 CFR 3.11 and 37 CFR 3.81(a). Completion of this form is NOT a substitute for filing an assignment.

(A) NAME OF ASSIGNEE

(B) RESIDENCE: (CITY and STATE OR COUNTRY)

Please check the appropriate assignee category or categories (will not be printed on the patent) : ☐ Individual ☐ Corporation or other private group entity ☐ Government

4a. Fees submitted: ☒ Issue Fee ☐ Publication Fee (if required) ☐ Advance Order - # of Copies ______

4b. Method of Payment: *(Please first reapply any previously paid fee shown above)*

☒ Electronic Payment via EFS-Web ☐ Enclosed check ☐ Non-electronic payment by credit card (Attach form PTO-2038)

☐ The Director is hereby authorized to charge the required fee(s), any deficiency, or credit any overpayment to Deposit Account No. ______

5. **Change in Entity Status** (from status indicated above)

☐ Applicant certifying micro entity status. See 37 CFR 1.29

☐ Applicant asserting small entity status. See 37 CFR 1.27

☐ Applicant changing to regular undiscounted fee status.

NOTE: Absent a valid certification of Micro Entity Status (see forms PTO/SB/15A and 15B), issue fee payment in the micro entity amount will not be accepted at the risk of application abandonment.
NOTE: If the application was previously under micro entity status, checking this box will be taken to be a notification of loss of entitlement to micro entity status.
NOTE: Checking this box will be taken to be a notification of loss of entitlement to small or micro entity status, as applicable.

NOTE: This form must be signed in accordance with 37 CFR 1.31 and 1.33. See 37 CFR 1.4 for signature requirements and certifications.

Authorized Signature /Jasmin Cosic/ Date August 16, 2019

Typed or printed name Jasmin Cosic Registration No. ______

Electronic Patent Application Fee Transmittal

Application Number:	15340991
Filing Date:	02-Nov-2016
Title of Invention:	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION
First Named Inventor/Applicant Name:	Jasmin Cosic
Filer:	Jasmin Cosic
Attorney Docket Number:	

Filed as Small Entity

Filing Fees for **Utility under 35 USC 111(a)**

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Basic Filing:				
Pages:				
Claims:				
Miscellaneous-Filing:				
Petition:				
Patent-Appeals-and-Interference:				
Post-Allowance-and-Post-Issuance:				
UTILITY APPL ISSUE FEE	2501	1	500	500

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Extension-of-Time:				
Miscellaneous:				
		Total in USD ($)		**500**

Electronic Acknowledgement Receipt

EFS ID:	36903735
Application Number:	15340991
International Application Number:	
Confirmation Number:	1993
Title of Invention:	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION
First Named Inventor/Applicant Name:	Jasmin Cosic
Customer Number:	116094
Filer:	Jasmin Cosic
Filer Authorized By:	
Attorney Docket Number:	
Receipt Date:	16-AUG-2019
Filing Date:	02-NOV-2016
Time Stamp:	18:14:30
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	yes
Payment Type	CARD
Payment was successfully received in RAM	$500
RAM confirmation Number	E20198FI18163567
Deposit Account	
Authorized User	

The Director of the USPTO is hereby authorized to charge indicated fees and credit any overpayment as follows:

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1	Issue Fee Payment (PTO-85B)	PTOL-85B_Issue_Fee_Form.pdf	87585 25f61878f8d47c7d0023a3252e8d625208f8230f	no	1
Warnings:					
Information:					
2	Fee Worksheet (SB06)	fee-info.pdf	29903 a1443dbe45e9b7975c9904ab21d1f5d6b0023dd8	no	2
Warnings:					
Information:					
		Total Files Size (in bytes):	117488		

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.
National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.
New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	ISSUE DATE	PATENT NO.	ATTORNEY DOCKET NO.	CONFIRMATION NO.
15/340,991	10/22/2019	10452974		1993

116094 7590 10/02/2019

Jasmin Cosic
108 Woodbury Street
Pawtucket, RI 02861

ISSUE NOTIFICATION

The projected patent number and issue date are specified above.

Determination of Patent Term Adjustment under 35 U.S.C. 154 (b)
(application filed on or after May 29, 2000)

The Patent Term Adjustment is 525 day(s). Any patent to issue from the above-identified application will include an indication of the adjustment on the front page.

If a Continued Prosecution Application (CPA) was filed in the above-identified application, the filing date that determines Patent Term Adjustment is the filing date of the most recent CPA.

Applicant will be able to obtain more detailed information by accessing the Patent Application Information Retrieval (PAIR) WEB site (http://pair.uspto.gov).

Any questions regarding the Patent Term Extension or Adjustment determination should be directed to the Office of Patent Legal Administration at (571)-272-7702. Questions relating to issue and publication fee payments should be directed to the Application Assistance Unit (AAU) of the Office of Data Management (ODM) at (571)-272-4200.

APPLICANT(s) (Please see PAIR WEB site http://pair.uspto.gov for additional applicants):

Jasmin Cosic, Miami, FL;

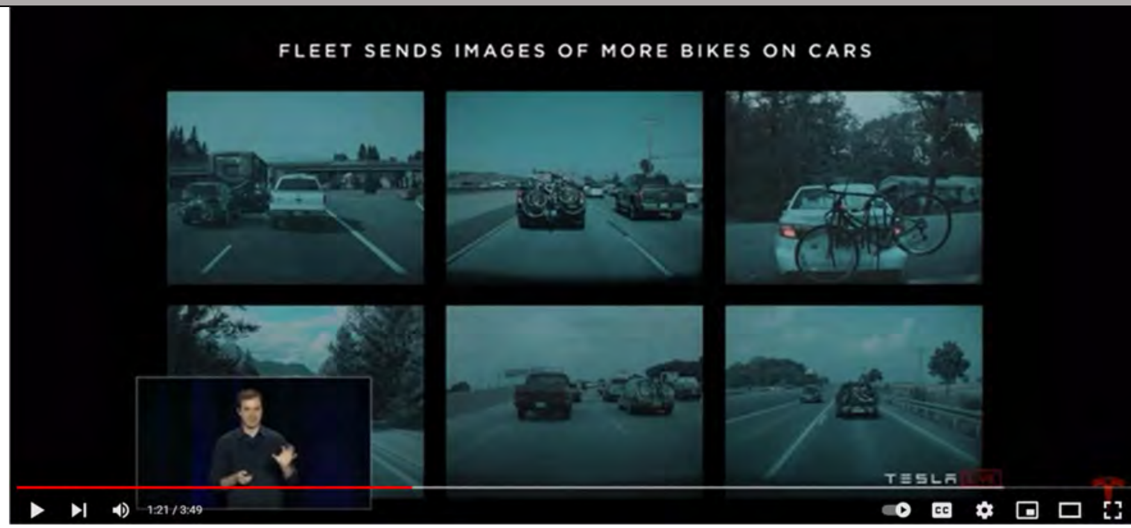
Exhibit H

U.S. Patent No. 10,452,974	
Claim 1	**Exemplary Infringement Evidence[1]**
[1pre] A method implemented using a computing system that includes one or more processor circuits, the method comprising:	Each autonomous Tesla vehicle with Software Version 9.0 and beyond (this includes vehicles with enhanced autopilot and/or full self-driving (FSD)) alone or together with the Dojo super computer infringes the '974 patent. Each of these vehicles practices a method of using a computing system that includes one or more processor circuits. A fleet of Tesla vehicles alone or together with the Tesla DoJo super computer is also an example of a computing system that practices the claimed method. As a driver drives a Tesla vehicle, the vehicle learns various circumstances (e.g., pedestrians, other vehicles, roads, buildings, bikes, road debris, animals, etc.) as well as driving instructions (e.g., instructions for effecting speed, steering, breaking, trajectory, etc.) that the driver used to navigate those circumstances. This driving knowledge learned on one Tesla vehicle is transmitted to the Tesla DoJo computer and then distributed to all vehicles in Tesla fleet via over-the-air (OTA) software updates. The fleet, therefore, enables each vehicle in the fleet to autonomously implement driving instructions learned on one or more originating vehicles when similar circumstances are detected by any of the vehicles in Tesla fleet, thereby enabling their autonomous driving. Tesla explains its object detection (e.g., circumstance representation) technology as follows: 

[1] These infringement contentions are prepared with publicly available information.

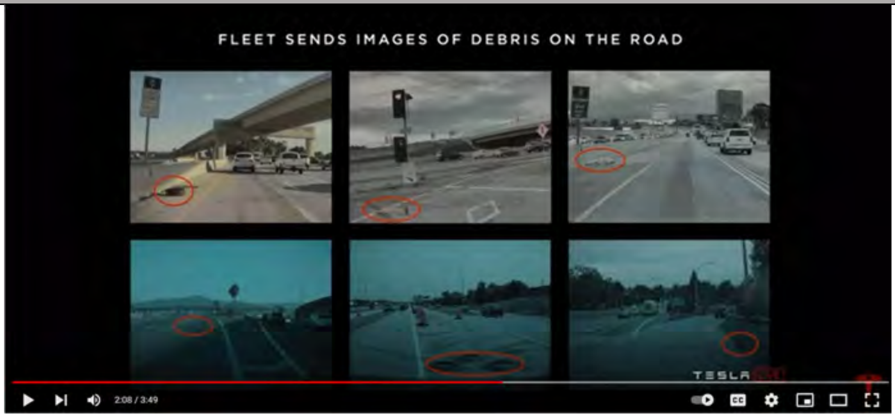
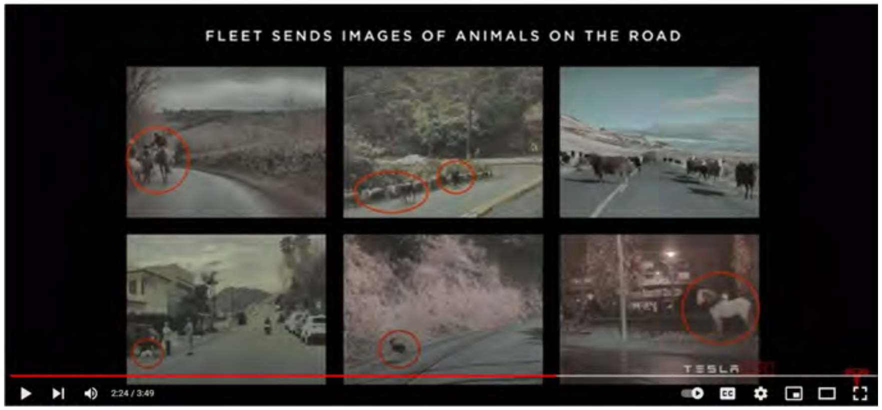
U.S. Patent No. 10,452,974	
	https://www.youtube.com/watch?v=33K3id2xNAE&t "so object detection is something we care a lot about we'd like to put bounding boxes around say the cars and the objects here because we need to track them and we need to understand how they might move around so again we might ask human annotators to give us some annotations for these and humans might go in and might tell you that ok those patterns over there are cars and bicycles and so on and you can train your neural network on this but if you're not careful the neural network all will make miss predictions in some cases"  https://www.youtube.com/watch?v=33K3id2xNAE&t "so as an example if we stumble by a car like this that has a bike on the back of it then the neural network actually when I joined would actually create two deductions it would create a car deduction and a bicycle deduction and that's actually kind of correct because I guess both of those objects actually exist but for the purposes of the controller and a planner downstream you really don't want to deal with the fact that this bicycle can go with the car the truth is that that bike is attached to that car so in terms of like just objects on the road there's a single object a single car and so what you'd like to do now is you'd like to just potentially annotate lots of those images as this is just a single car so the process that we that we go through internally in the team is that we take this image or a few images that show this pattern and *we have a mechanism a machine learning mechanism by which we can ask the fleet to source us examples that look like that and the fleet might respond with images that contains those patterns"*

U.S. Patent No. 10,452,974

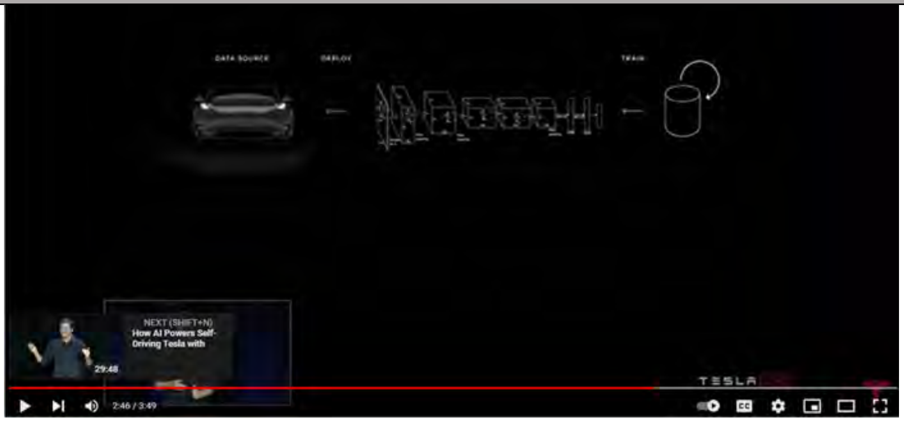


"so as an example these six images might come from the fleet they all contain bikes on backs of cars and we would go in and we would annotate all those as just a single car and then the the performance of that detector actually improves and the network internally understands that hey when the bike is just attached to the car that's actually just a single car and it can learn that given enough examples and that's how we sort of fix that problem … now the fleet doesn't just respond with bicycles on backs of cars we look for all the thing we look for lots of things all the time."

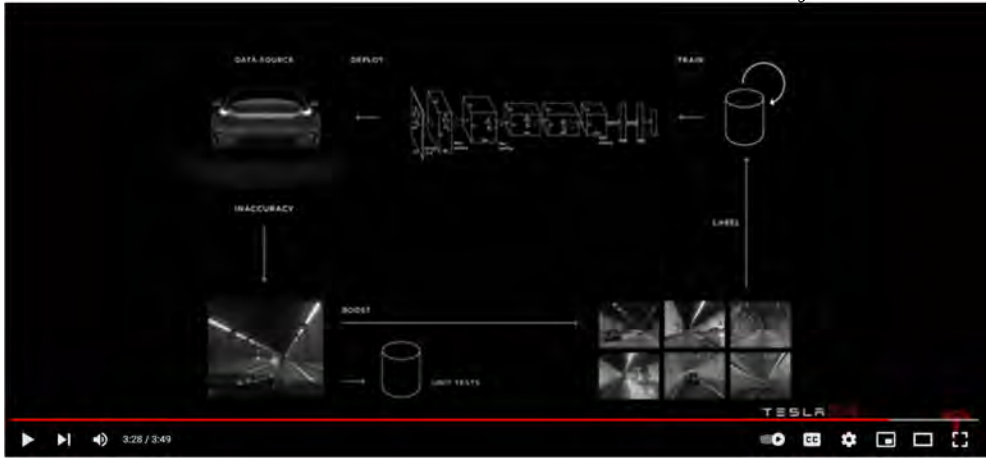
"so for example we look for boats and the fleet can respond with boats we look for construction sites and the fleet can send us lots of construction sites from across the world we look for even slightly more rare cases"

U.S. Patent No. 10,452,974	
	 "so for example finding debris on the road is pretty important to us so these are examples of images that have streamed to us from the fleet that show tires cones, plastic bags and things like that if we can source these at scale we can annotate them correctly and *the neural network will learn how to deal with them in the world*"  "here's another example animals of course also a very rare occurrence an event but we wanted the neural network to really understand what's going on here that these are animals and we want to deal with that correctly"

U.S. Patent No. 10,452,974



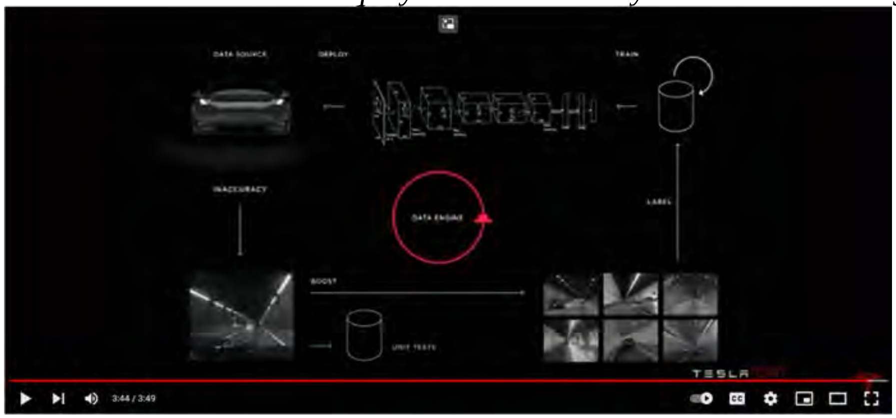
"so to summarize the process by which we iterate on neural network predictions looked something like this we start with a seed data set that was potentially sourced at random we annotate that data set and then we train your networks on that data set and put that in the car and then we have mechanisms by which we notice inaccuracies in the car when this detector may be misbehaving"



"so for example if we detect that the neural network might be uncertain or if we detect that or *if there's a driver intervention or any of those settings we can create this trigger infrastructure that sends us data of those inaccuracies and so for example if we don't perform very well on lane line detection on tunnels then we can notice that there's a problem in tunnels that image would enter our unit tests so we can verify that we've actually fixing the problem over time but now what you do is to fix this inaccuracy you*

U.S. Patent No. 10,452,974

need to source many more examples that look like that so we asked the fleet to please send us many more tunnels and then we label all those tunnels correctly we incorporate that into the training set and we retrain the network redeploy and iterate the cycle over and over again and



"so we refer to this iterative process by which we improve these predictions as the *data engine so iteratively deploying something potentially in shadow mode sourcing inaccuracies and incorporating the training set over and over again and we do this basically for all the predictions of these neural networks*"

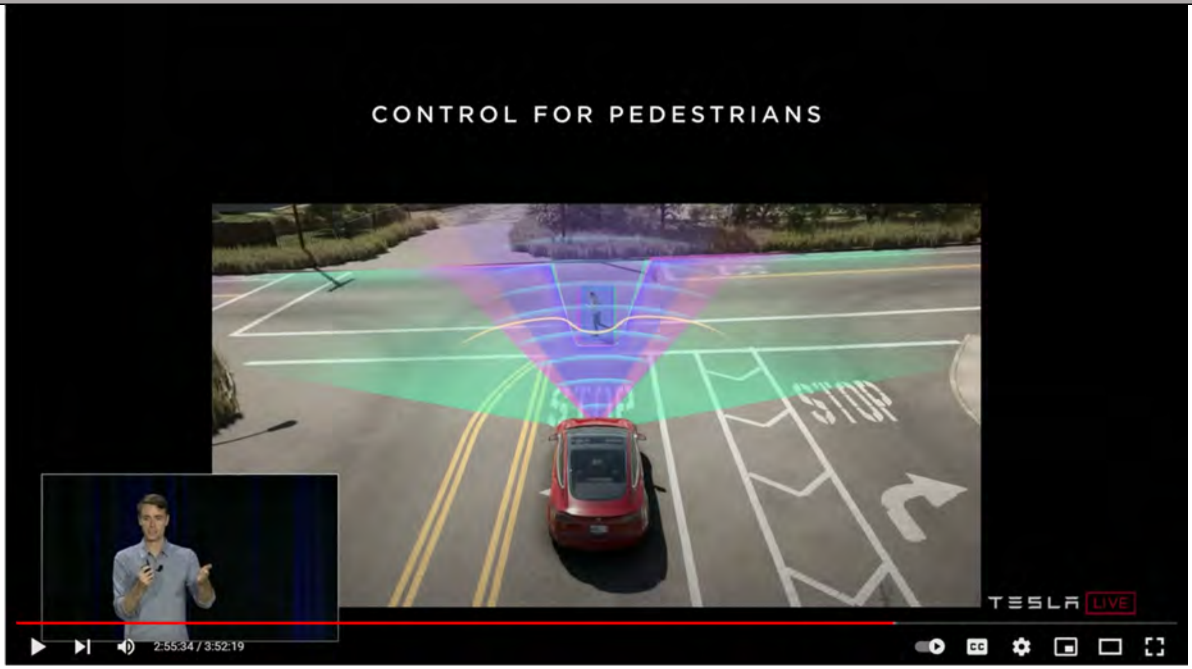
As Mr. Musk explained during Tesla AI Day all the human drivers are essentially training the neural net as to what is the correct course of action. See Tesla AI Day 2021 video https://www.youtube.com/watch?v=j0z4FweCy4M at 2:55:29. This is an example of the system being trained with object representations and instructions from a vehicle.

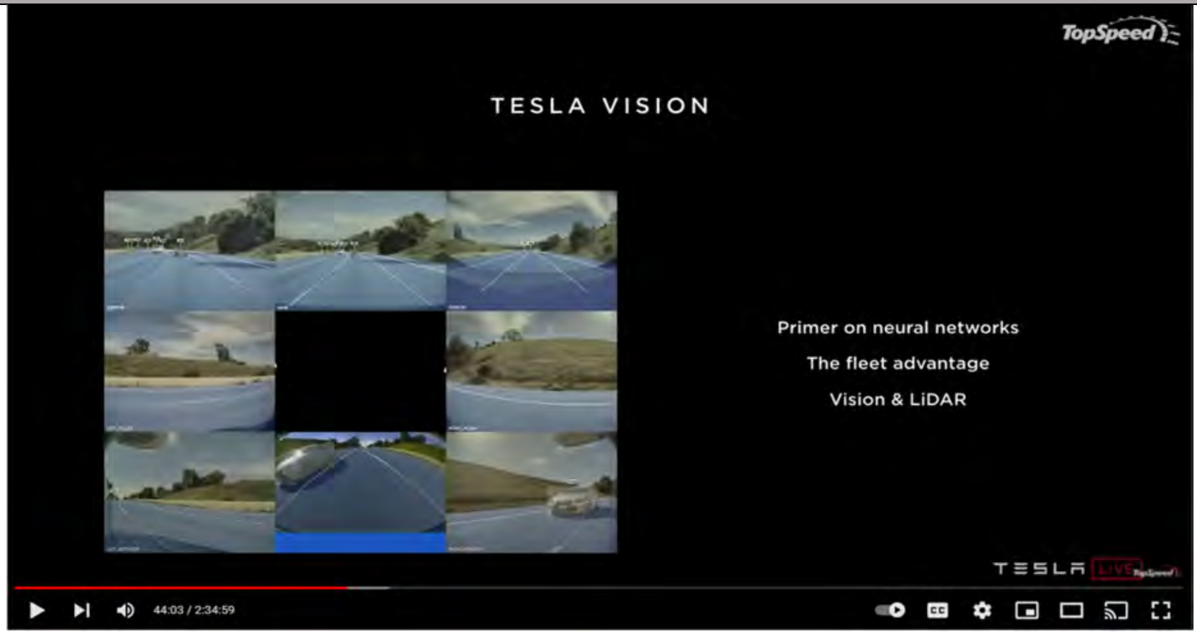
As Mr. Karpathy explained during Tesla Autonomy Day 2019 "while you are driving a car what you're actually doing is you are annotating the data because you are steering the wheel. you're telling us how to traverse different environments so what we're looking at here is a some person in the fleet who took a left through an intersection and what we do here is we have the full video of all the cameras and we know that the path that this person took because of the GPS, the inertial measurement unit, the wheel angle, the wheel ticks, so we put all that together and we understand the path that this person took through this environment and then of course we can use this for supervision [e.g., training a CNN through supervised learning] for the network so we just source a lot of this from the fleet, we train a

U.S. Patent No. 10,452,974	
	neural network on those trajectories, and then the neural network predicts paths just from that data so really what ***this is referred to typically is called imitation learning we're taking human trajectories from the real world I'm just trying to imitate how people drive in real worlds and we can also apply the same data engine crank to all of this and make this work.***" https://www.youtube.com/watch?v=-b041NXGPZ8 at 1:04:10. In this way, driving knowledge is: (i) learned on a first Tesla vehicle, (ii) transferred to the DoJo super computer where it is synthesized with other driving knowledge, (iii) packaged and sent from the DoJo to the fleet, e.g., the driving knowledge is stored on a memory of a second Tesla vehicle via the fleet over-the-air (OTA) software update, and (iv) accessed on the second Tesla vehicle for its autonomous driving. The second Tesla vehicle can, thereby, implement the driving knowledge learned on the first Tesla vehicle and synthesized on the DoJo. Each of the accused Tesla vehicles (Models 3, S, X, Y, etc.) includes one or more processors (e.g., the full self-driving computer/chip) configured to perform full self-driving.   See Tesla Autonomy Day 2019 video https://www.youtube.com/watch?v=-b041NXGPZ8 at 7:11 (Tesla full self driving computer) and at 10:22 (Tesla full self driving chip).
[1a] accessing a memory that stores at least a knowledgebase that includes: a first correlation	The processor of the second Tesla vehicle **(the claimed "second device")** accesses its memory that stores at least a knowledgebase that includes **a first circumstance representation** (e.g., representation of pedestrians, other vehicles, roads, buildings, etc.) correlated with a first set of driving instructions (e.g., driving instructions for effecting speed, steering, breaking, trajectory, etc.; **the claimed "first one or more instruction sets"**) for operating a first Tesla vehicle **(the claimed "first device") and a second circumstance representation** (e.g., representation of pedestrians, other vehicles, roads,

U.S. Patent No. 10,452,974	
including a first circumstance representation correlated with a first one or more instruction sets for operating a first device and a second correlation including a second circumstance representation correlated with a second one or more instruction sets for operating the first device, wherein the first circumstance representation represents a first circumstance detected at least in part by one or more sensors of the first device and the second circumstance representation represents a second circumstance detected at least in part by the one or more sensors of the first device, and wherein at least a portion of the first correlation and	buildings, etc.) correlated with a second set of driving instructions (e.g., driving instructions for effecting speed, steering, breaking, trajectory, etc.; **the claimed "second one or more instruction sets"**) for operating a first Tesla vehicle.  See Tesla AI Day 2021 video https://www.youtube.com/watch?v=j0z4FweCy4M at 1:31:58 (example of the claimed "circumstance representation")

U.S. Patent No. 10,452,974	
at least a portion of the second correlation are learned in a learning process that includes operating the first device at least partially by a user;	 See Tesla AI Day 2021 video https://www.youtube.com/watch?v=j0z4FweCy4M at 1:32:40-48 (combining everything together we can produce these amazing data sets that annotate all of the road texture or the static objects and all of the moving objects **[example of the claimed examples of first and second "circumstance representation[s]"]**)

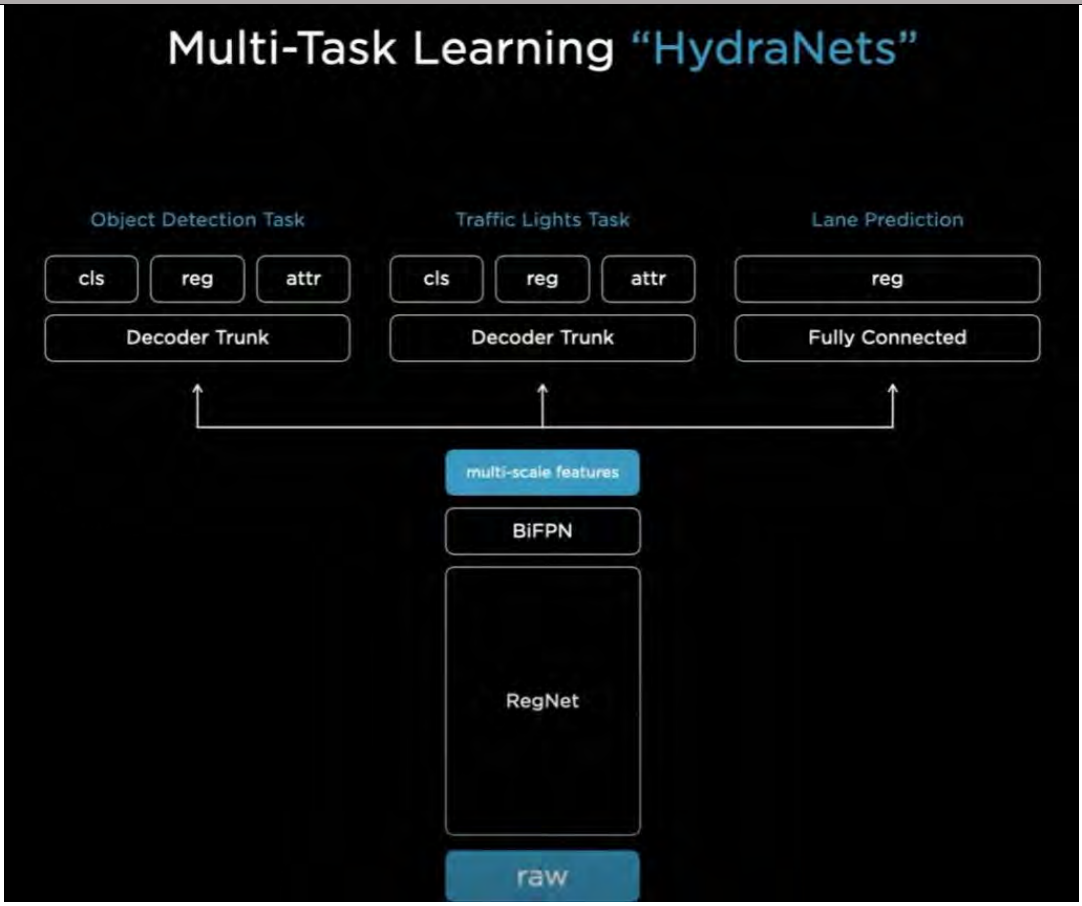
U.S. Patent No. 10,452,974	
	 See Tesla AI Day 2021 video https://www.youtube.com/watch?v=j0z4FweCy4M at 2:55:34 (discussing an automatic emergency breaking system). As shown in the clip, each Tesla vehicle has sensors such as cameras, etc. that are used to detect a first circumstance, e.g., a pedestrian in front of the vehicle.

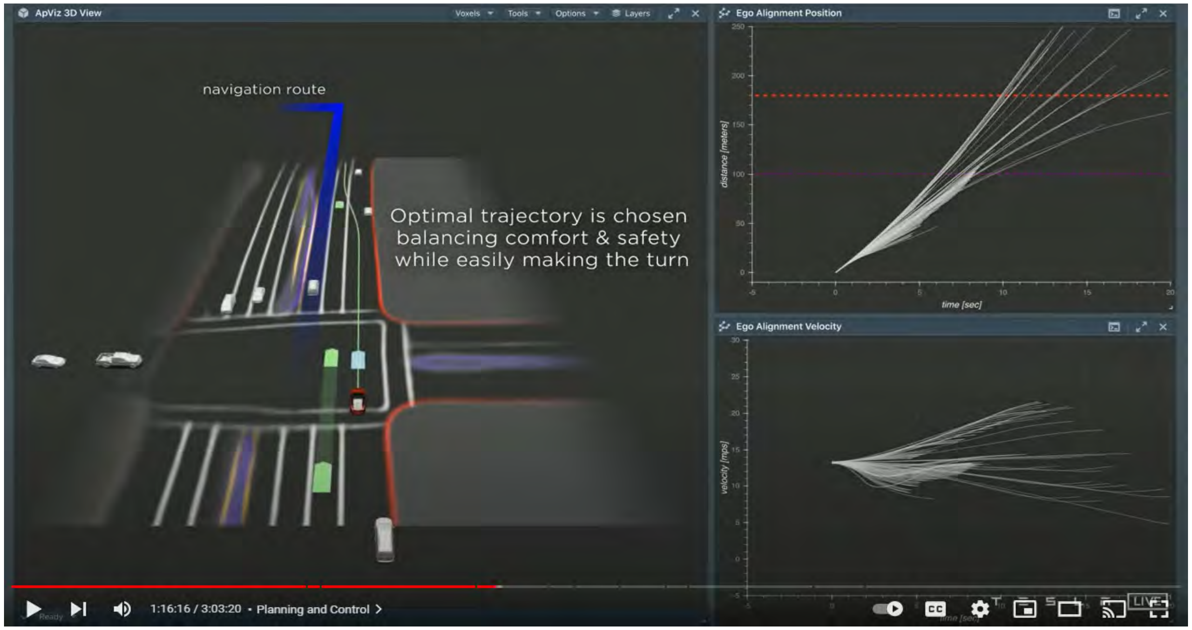
U.S. Patent No. 10,452,974	
	 *See also* Tesla Autonomy Day 2019 video https://www.youtube.com/watch?v=-b041NXGPZ8 at 44:00 (a stream of videos from eight cameras **[the claimed "one or more sensors"]** across the vehicle used to make a lane change). As detailed above, the circumstance representations (e.g., a representation of a pedestrian in front the vehicle, representation of surrounding vehicles in a lane change situation, animals or debris in the road, a stop light, a stop sign, etc.) **("wherein the first circumstance representation represents a first circumstance detected at least in part by one or more sensors of the first device and the second circumstance representation represents a second circumstance detected at least in part by the one or more sensors of the first device")** and the corresponding driving instructions (e.g., instructions for applying the breaks so the pedestrian is not hit or so that traffic laws are obeyed, or instructions for turning the wheel to safely change lanes or avoid an animal or debris, etc.) are learned in a process that involves a driver **(the claimed "user")** operating the first Tesla vehicle **(the claimed "wherein at least a portion of the first correlation and at least a portion of the second correlation are learned in a learning process that includes**

U.S. Patent No. 10,452,974	
	operating the first device at least partially by a user”).  See Tesla Autonomy Day 2019 video https://www.youtube.com/watch?v=-b041NXGPZ8 at 1:04:10 (while you **[the claimed “user”]** are driving a car what you’re actually doing is you are ***annotating the data because you are steering the wheel*** **[e.g., learned instructions]** you’re telling us how to traverse different environments so what we’re looking at here is some person in the fleet who took a left through an intersection and what we do here is we have the full video of all the cameras and we know that the path that this person took because of the GPS, the inertial measurement unit, the wheel angle, the wheel ticks, so we put all that together and we understand the path that this person took through this environment **[circumstance representation representing the environment correlated with driving instructions on how to traverse the environment]**. and then of course we can use this for supervision for the network so we just source a lot of this from the fleet, we train a neural network on those trajectories, and then the neural network predicts paths just from that data. … we’re taking human trajectories from the real world we’re just trying to imitate how people drive in real worlds) See Tesla AI Day 2021 video https://www.youtube.com/watch?v=j0z4FweCy4M at 2:55:29 (all the

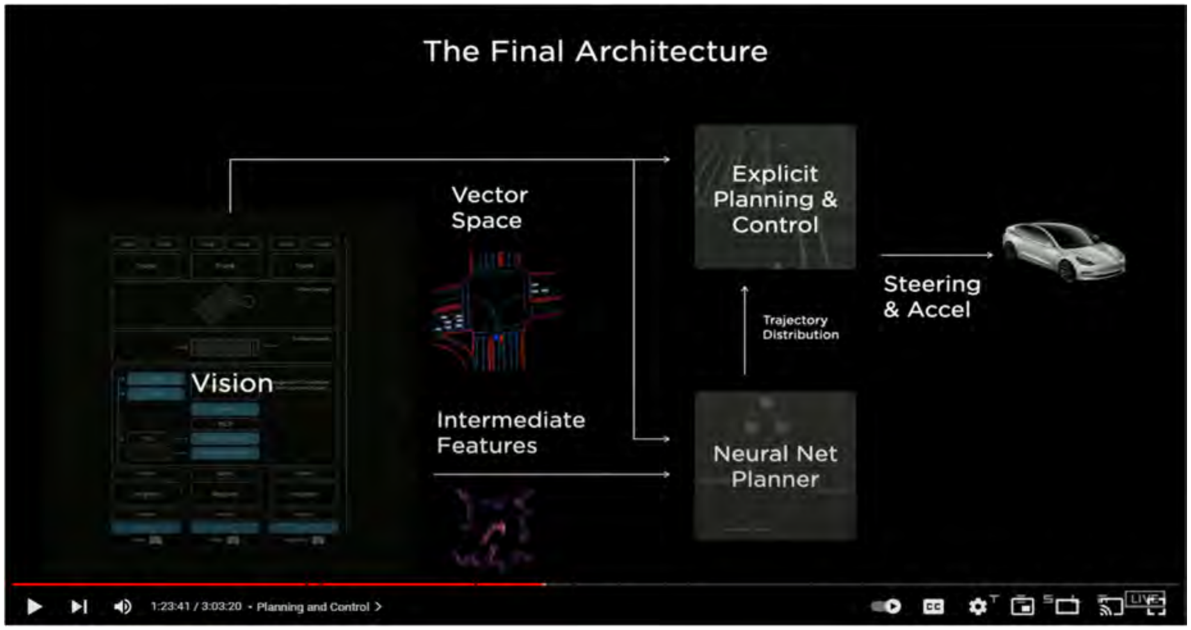
U.S. Patent No. 10,452,974	
	human drivers are essentially training the neural net as to what is the correct course of action **[the claimed driving instructions]**)  *See* Musk, Karpathy, etc. in Tesla Autonomy Day 2019 video at https://www.youtube.com/watch?v=-b041NXGPZ8 at 1:03:50 (everyone is training the network all the time), 1:30:55 (Tesla system learns from drivers training the system), and 1:52:23 (high level chart clearly showing monitoring and learning from human drivers and implementing autonomous driving based on the learned knowledge).
[1b] generating or receiving a third circumstance representation, wherein the third circumstance representation represents a third circumstance	The processor of the second Tesla vehicle **(the claimed "second device")** generates a **third circumstance representation** that represents a current circumstance (e.g., pedestrians, other vehicles, roads, buildings, animals, debris, etc.) detected at least in part by its cameras **(the claimed "one or more sensors of a second device")**. For instance, when the system is deployed to the fleet, a second Tesla vehicle **(the claimed "second device")** generates a **third circumstance representation** (e.g., representation of a different pedestrian

U.S. Patent No. 10,452,974	
detected at least in part by the one or more sensors of the first device ***or at least in part by one or more sensors of a second device***;	in front the vehicle, representation of different surrounding vehicles in a lane change situation, representation of a different intersection in a left turn situation, etc.) that is detected by the second Tesla vehicle's sensors (e.g. cameras). The **third circumstance representation** is generated so it can be compared with previously learned circumstance representations **(the claimed "first circumstance representation" or "second circumstance representation")**, and if a similar match **(the claimed "at least partial match" in claim element 1c)** is found, the correlated driving instructions **(the claimed "first one or more instruction sets" or "second one or more instruction sets" in claim element 1a)** can be used for autonomous driving. Analogous evidence as for **first and second circumstance representations** and **"one or more sensors"** recited in claim element 1a applies to **"third circumstance representation"** and **"one or more sensors"** here (e.g., information detected by the second car's sensors), and it is not duplicated to save space. See claim element 1c. As explained during the 2021 AI day, Tesla said it "discovered that we don't just want to detect cars, we want to do a large number of tasks. So for example, we want to do traffic light recognition and detection, a lane prediction and so on. So very quickly, we conversioned this kind of architectural layout, where there's a common shared backbone, and then branches off into a number of heads. So we call these therefore Hydra Nets. And these are the heads of the Hydra." https://elon-musk-interviews.com/2021/08/31/tesla-ai-day-the-presentation-i/

U.S. Patent No. 10,452,974	
	 https://elon-musk-interviews.com/2021/08/31/tesla-ai-day-the-presentation-i/
[1c] anticipating the first one or more instruction sets for operating the first device learned in the learning process based on at least partial match between the third	The processor of the second Tesla vehicle **(the claimed "second device")** anticipates a set of driving instructions (e.g., instructions for applying the breaks so the pedestrian is not hit, instructions for turning the wheel to safely change lanes or to avoid an animal or debris, instructions for turning left in an intersection, etc.; **the claimed "first one or more instruction sets"**) based on similarity **(the claimed "at least partial match")** between the **first circumstance representation** (e.g., a representation of a pedestrian in front the vehicle, a representation of surrounding vehicles in a lane change situation, a representation of an animal or debris, representation of an intersection in a left turn

U.S. Patent No. 10,452,974	
circumstance representation and the first circumstance representation; and	situation, etc. as previously learned on the first Tesla vehicle) and a **third circumstance representation** (e.g., a representation of a pedestrian in front the vehicle, a representation of surrounding vehicles in a lane change situation, a representation of an animal or debris, representation of an intersection in a left turn situation, etc. as currently detected by the second Tesla vehicle). Therefore, Tesla vehicles anticipate previously learned driving instructions based on similarity between the currently generated circumstance representation and previously learned circumstance representations.  See Tesla AI Day 2021 video https://www.youtube.com/watch?v=j0z4FweCy4M at 1:15:26 (let's see an example of how the search **[e.g., search for previously learned driving instructions for effecting a trajectory correlated with a previously learned circumstance representation that at least partially matches the incoming circumstance representation]** operates. so here we're trying to do a lane change. in this case the car needs to do two back to back lane changes to make the left turn up ahead. for this, the car searches over different maneuvers. so the first one it searches is a lane change that's close by but the car breaks pretty harshly so it's pretty uncomfortable. the next maneuver it tries

U.S. Patent No. 10,452,974	
	does the lane change a bit late so it speeds up, goes beyond the other car, goes in front of the other cars, and find it at the lane change but now it risks missing the left turn. we do thousands of such searches in a very short time span). *See also id.* at 2:32:42 (similarly, for planning, we need to bake in a search **[e.g., search for previously learned driving instructions for effecting a trajectory correlated with a previously learned circumstance representation that at least partially matches the incoming circumstance representation]** and optimization into the planning, into the network architecture, and once we do that we should be able to do planning very quickly).
[1d] at least in response to the anticipating, executing the first one or more instruction sets for operating the first device learned in the learning process, wherein the first device or the second device autonomously performs one or more operations defined by the first one or more instruction sets for operating the first device learned in the learning process.	The processor of the second Tesla vehicle **(the claimed "second device")** executes, at least in response to the anticipating (claim element 1d), the previously learned set of driving instructions **(the claimed "first one or more instruction sets for operating the first device")** so that the second vehicle **(the claimed "second device")** can drive autonomously based on the driving instructions learned on the first Tesla vehicle **(the claimed "first device")**. For instance, memory of the second Tesla vehicle stores a **first circumstance representation** (e.g., representation of a pedestrian in front the vehicle, representation of surrounding vehicles in a lane change situation, representation of an intersection in a left turn situation, etc.) correlated with a set of driving instructions (e.g., instructions for applying the breaks so the pedestrian is not hit, instructions for turning the wheel to safely change lanes, instructions for turning left in an intersection, etc.; **the claimed "first one or more instruction sets"**) that have previously been learned on a first Tesla vehicle **(the claimed "first device")** and have been transferred to the second Tesla vehicle **(the claimed "second device")** via the fleet over-the-air (OTA) software update. In response to determining that the incoming circumstance representation (e.g., representation of a pedestrian currently in front the vehicle, representation of currently surrounding vehicles in a lane change situation, representation of a current intersection in a left turn situation, etc.) is similar to **(the claimed "at least partial match")** the previously learned circumstance representation (e.g., representation of a pedestrian previously in front the vehicle, representation of previously surrounding vehicles in a lane change situation, representation of a previous intersection in a left turn situation, etc.), the processor of the second Tesla vehicle **(the claimed "second device")** causes the previously learned driving instruction (e.g., instructions for applying the breaks so the pedestrian is not hit, instructions for turning the wheel to safely change lanes, instructions for turning left in an intersection, etc.) to be executed. As stated by Elon Musk on Tesla Autonomy Day 2019, Tesla system distributes driving knowledge

U.S. Patent No. 10,452,974	
	learned from multiple Tesla vehicles to all Tesla vehicles via over-the-air (OTA) software updates, thereby enabling the second Tesla vehicle **(the claimed "second device")** to autonomously implement driving knowledge learned on the first vehicle **(the claimed "first device")** as claimed. Mr. Musk and Mr. Karpathy describe this as a crucial competitive advantage of Tesla over other autonomous driving companies. *See* Musk, Karpathy, etc. in Tesla Autonomy Day 2019 video https://www.youtube.com/watch?v=-b041NXGPZ8 at 38:58 ("I think a very powerful sustainable advantage for us is the fleet"), 44:40 ("it is such a big deal that we have the fleet"), 55:55 ("why is Tesla in such a unique and interesting position to really get all these three essentials right, and the answer to that of course is the fleet"), 122:14 ("it's extremely difficult to catch up when Tesla has 100 times more miles per day than everyone else combined").  See Tesla AI Day 2021 video https://www.youtube.com/watch?v=j0z4FweCy4M at 1:23:40 (so this is what a final architecture is going to look like. the vision system is going to crush down the dense video data into a vector space **[incoming circumstance representation representing objects in the second Tesla vehicle's surrounding]**. it's going to be consumed by both an explicit planner and a neural

U.S. Patent No. 10,452,974	
	network planner **[CNN that stores previously learned circumstance representations correlated with previously learned driving instructions and that searches for at least partially matching circumstance representation and correlated driving instructions]** in addition to this, the network planner can also consume intermediate features of the network. Together, this produces a trajectory distribution and it can be optimized end to end both with explicit cost functions and human intervention and other imitation data. this then goes into explicit planning function that does whatever it sees for that and produces the final steering and acceleration commands for the car).
Claim 14	**Exemplary Evidence**
14. The method of claim 1, wherein the first circumstance representation includes: one or more object representations, or one or more collections of object representations, and wherein the second circumstance representation includes: one or more object representations, or one or more collections of object representations, and wherein the third circumstance representation includes: one or more object representations, or one or more collections of object representations.	Each autonomous Tesla vehicle with Software Version 9.0 and beyond (this includes vehicles with enhanced autopilot and/or full self-driving (FSD)) alone or together with the Dojo super computer practices the method of claim 1. Also, the first circumstance representation includes: one or more object representations, or one or more collections of object representations. The second circumstance representation includes: one or more object representations, or one or more collections of object representations. And the third circumstance representation includes: one or more object representations, or one or more collections of object representations. For example, the processor of the Tesla vehicle accesses its memory that stores at least a knowledgebase that includes **first and second circumstance representations** each including **one or more object representations** or collections of object representations (e.g., representations of pedestrians, other vehicles, roads, buildings, and collections of such representations, e.g., a bike on the back of a car, for example, etc.). And the third circumstance representation received or generated by the Tesla vehicle includes one or more object representations (or one or more collections of object representations) (e.g., (e.g., representations of pedestrians, other vehicles, roads, buildings, and collections of such representations, e.g., a bike on the back of a car, for example, etc.).

U.S. Patent No. 10,452,974	
	 See Tesla AI Day 2021 video https://www.youtube.com/watch?v=j0z4FweCy4M at 1:31:58 (example of the claimed "circumstance representations" with multiple object representations).
Claim 18	**Exemplary Infringement Evidence**
[18pre] One or more non-transitory machine readable media storing machine readable code that when executed by one or more processor circuits causes the one or more processor circuits to perform at least:	Each autonomous Tesla vehicle with Software Version 9.0 and beyond (this includes vehicles with enhanced autopilot and/or full self-driving (FSD)) alone or together with the Dojo super computer comprises one or more non-transitory machine readable media storing machine readable code that when executed by one or more processor circuits causes the one or more processor circuits to perform the limitations below. The evidence cited for claim [1pre] above applies to this limitation and is incorporated by reference.
[18a] accessing a memory that stores at least a	Each autonomous Tesla vehicle with Software Version 9.0 and beyond (this includes vehicles with enhanced autopilot and/or full self-driving (FSD)) alone or together with the Dojo super computer

U.S. Patent No. 10,452,974	
knowledgebase that includes: a first correlation including a first circumstance representation correlated with a first one or more instruction sets for operating a first device and a second correlation including a second circumstance representation correlated with a second one or more instruction sets for operating the first device, wherein the first circumstance representation represents a first circumstance detected at least in part by one or more sensors of the first device and the second circumstance representation represents a second circumstance detected at least in part by the one or more sensors of the first device, and wherein at least a portion of the first correlation and at least a portion of the second correlation are	comprises machine readable code that can accessing a memory that stores at least a knowledgebase that includes: a first correlation including a first circumstance representation correlated with a first one or more instruction sets for operating a first device and a second correlation including a second circumstance representation correlated with a second one or more instruction sets for operating the first device, wherein the first circumstance representation represents a first circumstance detected at least in part by one or more sensors of the first device and the second circumstance representation represents a second circumstance detected at least in part by the one or more sensors of the first device, and wherein at least a portion of the first correlation and at least a portion of the second correlation are learned in a learning process that includes operating the first device at least partially by a user. The evidence cited for claim [1a] above applies to this limitation and is incorporated by reference.

U.S. Patent No. 10,452,974	
learned in a learning process that includes operating the first device at least partially by a user;	
[18b] generating or receiving a third circumstance representation, wherein the third circumstance representation represents a third circumstance detected at least in part by the one or more sensors of the first device or at least in part by one or more sensors of a second device;	Each autonomous Tesla vehicle with Software Version 9.0 and beyond (this includes vehicles with enhanced autopilot and/or full self-driving (FSD)) alone or together with the Dojo super computer comprises machine readable code that can at least in response to the anticipating, causing the first device or the second device to perform one or more operations defined by the first one or more instruction sets for operating the first device learned in the learning process at least by causing the one or more processor circuits or another one or more processor circuits to execute the first one or more instruction sets for operating the first device learned in the learning process. The evidence cited for claim [1b] above applies to this limitation and is incorporated by reference.
[18c] anticipating the first one or more instruction sets for operating the first device learned in the learning process based on at least partial match between the third circumstance representation and the first circumstance representation; and	Each autonomous Tesla vehicle with Software Version 9.0 and beyond (this includes vehicles with enhanced autopilot and/or full self-driving (FSD)) alone or together with the Dojo super computer comprises machine readable code that can anticipate the first one or more instruction sets for operating the first device learned in the learning process based on at least partial match between the third circumstance representation and the first circumstance representation. The evidence cited for claim [1c] above applies to this limitation and is incorporated by reference.
[18d] at least in response to the anticipating, causing the first device or the second device to perform	Each autonomous Tesla vehicle with Software Version 9.0 and beyond (this includes vehicles with enhanced autopilot and/or full self-driving (FSD)) alone or together with the Dojo super computer comprises machine readable code that can at least in response to the anticipating, causing the first device or the second device to perform one or more operations defined by the first one or more

U.S. Patent No. 10,452,974	
one or more operations defined by the first one or more instruction sets for operating the first device learned in the learning process at least by causing the one or more processor circuits or another one or more processor circuits to execute the first one or more instruction sets for operating the first device learned in the learning process.	instruction sets for operating the first device learned in the learning process at least by causing the one or more processor circuits or another one or more processor circuits to execute the first one or more instruction sets for operating the first device learned in the learning process. The evidence cited for claim [1d] above applies to this limitation and is incorporated by reference.

Exhibit I

U.S. Patent No. 11,238,344	
Claim 1	**Exemplary Infringement Evidence**[1]
[1pre] A system comprising:	Each autonomous Tesla vehicle with Software Version 9.0 and beyond (this includes vehicles with enhanced autopilot and/or full self-driving (FSD)) alone or together with the Dojo super computer infringes the '344 patent. Each of these vehicles is an example of a system that includes a processor, memory, etc. The Tesla fleet includes over a million of such customer vehicles. A fleet of Tesla vehicles alone or together with the Tesla DoJo super computer is also an example of a covered system. As a driver drives a Tesla vehicle, the vehicle learns various circumstances (e.g., pedestrians, other vehicles, roads, buildings, bikes, road debris, animals, etc.) as well as driving instructions (e.g., instructions for effecting speed, steering, breaking, trajectory, etc.) that the driver used to navigate those circumstances. This driving knowledge learned on one Tesla vehicle is transmitted to the Tesla DoJo computer and then distributed to all vehicles in Tesla fleet via over-the-air (OTA) software updates. The fleet, therefore, enables each vehicle in the fleet to autonomously implement driving instructions learned on one or more originating vehicles when similar circumstances are detected by any of the vehicles in Tesla fleet, thereby enabling their autonomous driving. Tesla explains its object detection (e.g., circumstance representation) technology as follows: 

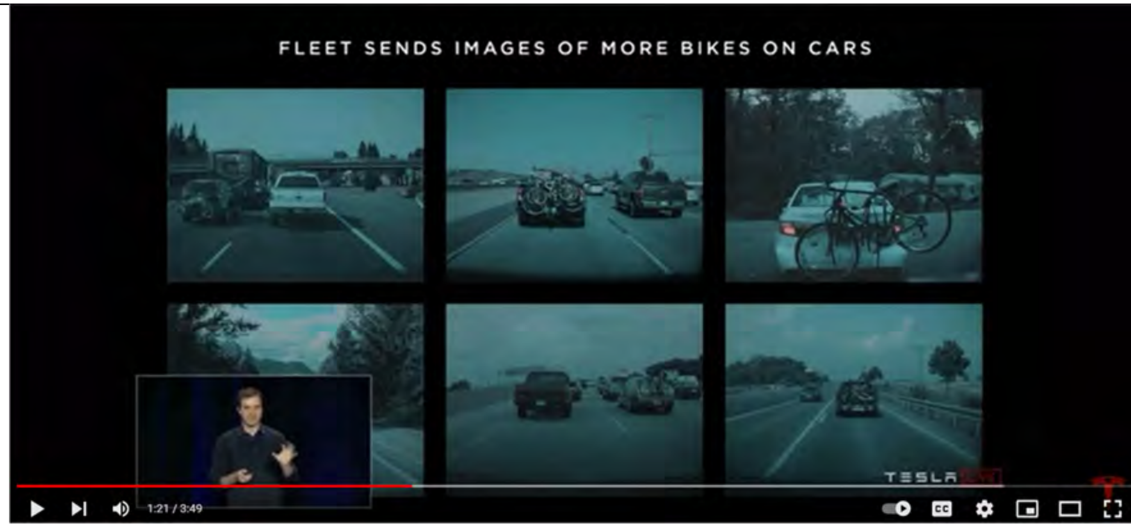
[1] These infringement contentions are prepared with publicly available information.

https://www.youtube.com/watch?v=33K3id2xNAE&t

"so object detection is something we care a lot about we'd like to put bounding boxes around say the cars and the objects here because we need to track them and we need to understand how they might move around so again we might ask human annotators to give us some annotations for these and humans might go in and might tell you that ok those patterns over there are cars and bicycles and so on and you can train your neural network on this but if you're not careful the neural network all will make miss predictions in some cases"

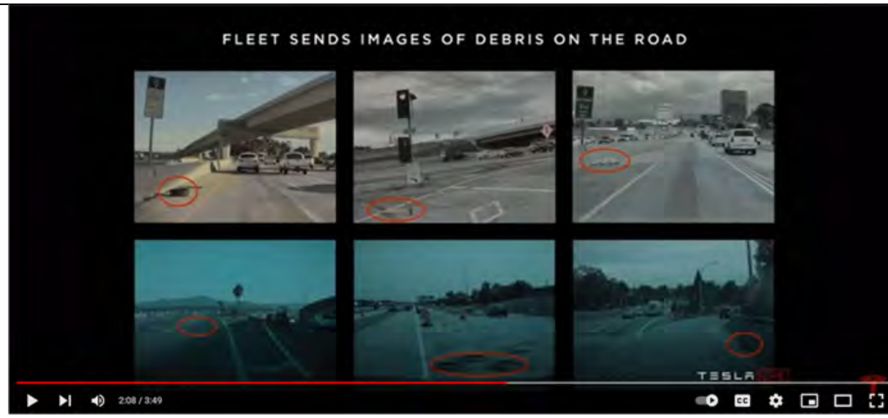
https://www.youtube.com/watch?v=33K3id2xNAE&t

"so as an example if we stumble by a car like this that has a bike on the back of it then the neural network actually when I joined would actually create two deductions it would create a car deduction and a bicycle deduction and that's actually kind of correct because I guess both of those objects actually exist but for the purposes of the controller and a planner downstream you really don't want to deal with the fact that this bicycle can go with the car the truth is that that bike is attached to that car so in terms of like just objects on the road there's a single object a single car and so what you'd like to do now is you'd like to just potentially annotate lots of those images as this is just a single car so the process that we that we go through internally in the team is that we take this image or a few images that show this pattern and *we have a mechanism a machine learning mechanism by which we can ask the fleet to source us examples that look like that and the fleet might respond with images that contains those patterns"*

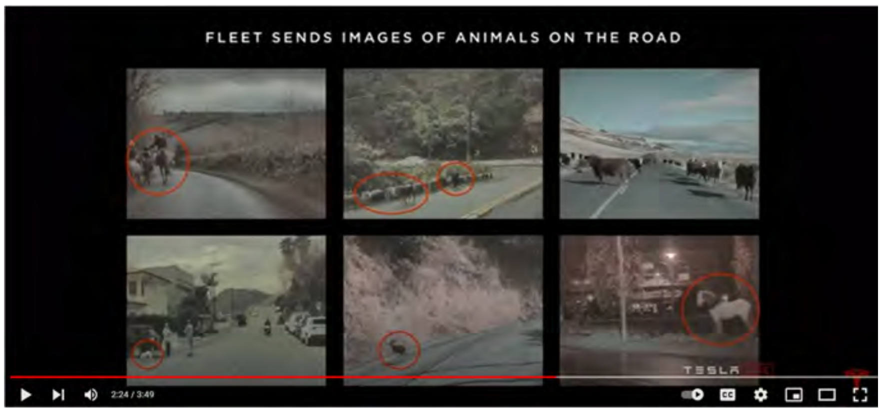


"so as an example these six images might come from the fleet they all contain bikes on backs of cars and we would go in and we would annotate all those as just a single car and then the the performance of that detector actually improves and the network internally understands that hey when the bike is just attached to the car that's actually just a single car and it can learn that given enough examples and that's how we sort of fix that problem … now the fleet doesn't just respond with bicycles on backs of cars we look for all the thing we look for lots of things all the time."

"so for example we look for boats and the fleet can respond with boats we look for construction sites and the fleet can send us lots of construction sites from across the world we look for even slightly more rare cases"

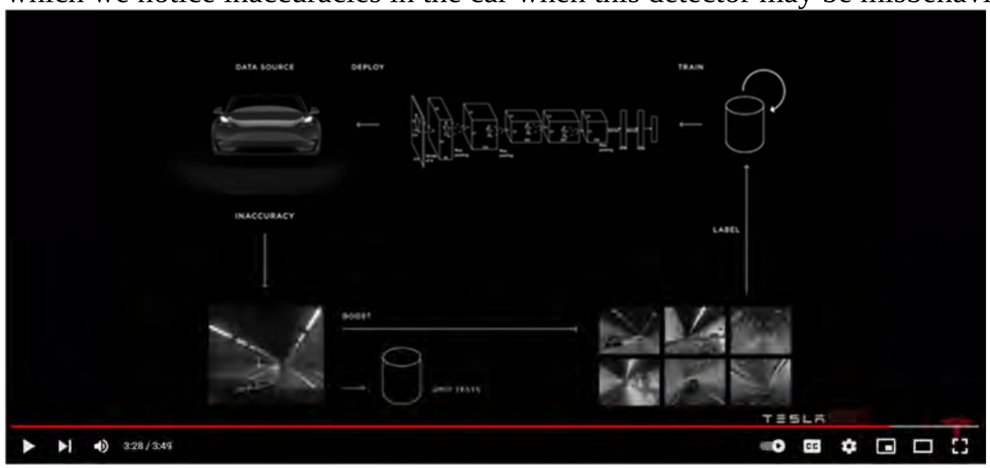


"so for example finding debris on the road is pretty important to us so these are examples of images that have streamed to us from the fleet that show tires cones, plastic bags and things like that if we can source these at scale we can annotate them correctly and *the neural network will learn how to deal with them in the world*"



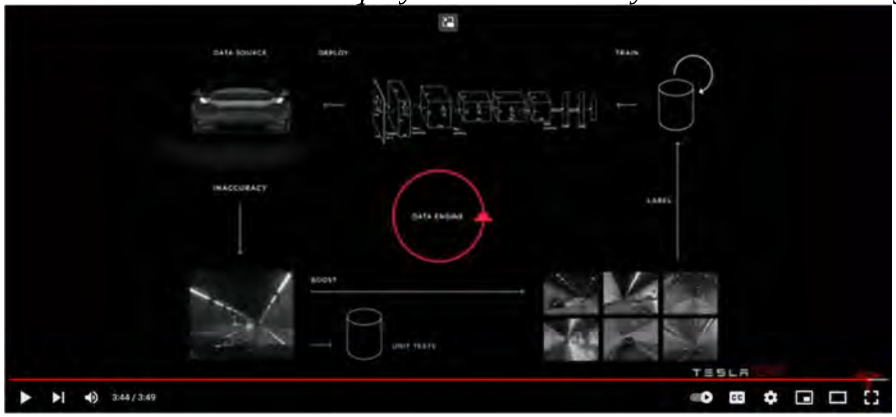
"here's another example animals of course also a very rare occurrence an event but we wanted the neural network to really understand what's going on here that these are animals and we want to deal with that correctly"

"so to summarize the process by which we iterate on neural network predictions looked something like this we start with a seed data set that was potentially sourced at random we annotate that data set and then we train your networks on that data set and put that in the car and then we have mechanisms by which we notice inaccuracies in the car when this detector may be misbehaving"



"so for example if we detect that the neural network might be uncertain or if we detect that or *if there's a driver intervention or any of those settings we can create this trigger infrastructure that sends us data of those inaccuracies and so for example if we don't perform very well on lane line detection on tunnels then we can notice that there's a problem in tunnels that image would enter our unit tests so we can verify that we've actually fixing the problem over time but now what you do is to fix this inaccuracy you need to source many more examples that look like that so we asked the fleet to please send us many*

more tunnels and then we label all those tunnels correctly we incorporate that into the training set and we retrain the network redeploy and iterate the cycle over and over again and



"so we refer to this iterative process by which we improve these predictions as the *data engine so iteratively deploying something potentially in shadow mode sourcing inaccuracies and incorporating the training set over and over again and we do this basically for all the predictions of these neural networks*"

As Mr. Musk explained during Tesla AI Day all the human drivers are essentially training the neural net as to what is the correct course of action. See Tesla AI Day 2021 video https://www.youtube.com/watch?v=j0z4FweCy4M at 2:55:29. This is an example of the system being trained with object representations and instructions from a vehicle.

As Mr. Karpathy explained during Tesla Autonomy Day 2019 "while you are driving a car what you're actually doing is you are annotating the data because you are steering the wheel. you're telling us how to traverse different environments so what we're looking at here is a some person in the fleet who took a left through an intersection and what we do here is we have the full video of all the cameras and we know that the path that this person took because of the GPS, the inertial measurement unit, the wheel angle, the wheel ticks, so we put all that together and we understand the path that this person took through this environment and then of course we can use this for supervision [e.g., training a CNN through supervised learning] for the network so we just source a lot of this from the fleet, we train a neural network on those trajectories, and then the neural network predicts paths just from that data so really what ***this is referred to typically is called imitation learning we're taking human trajectories***

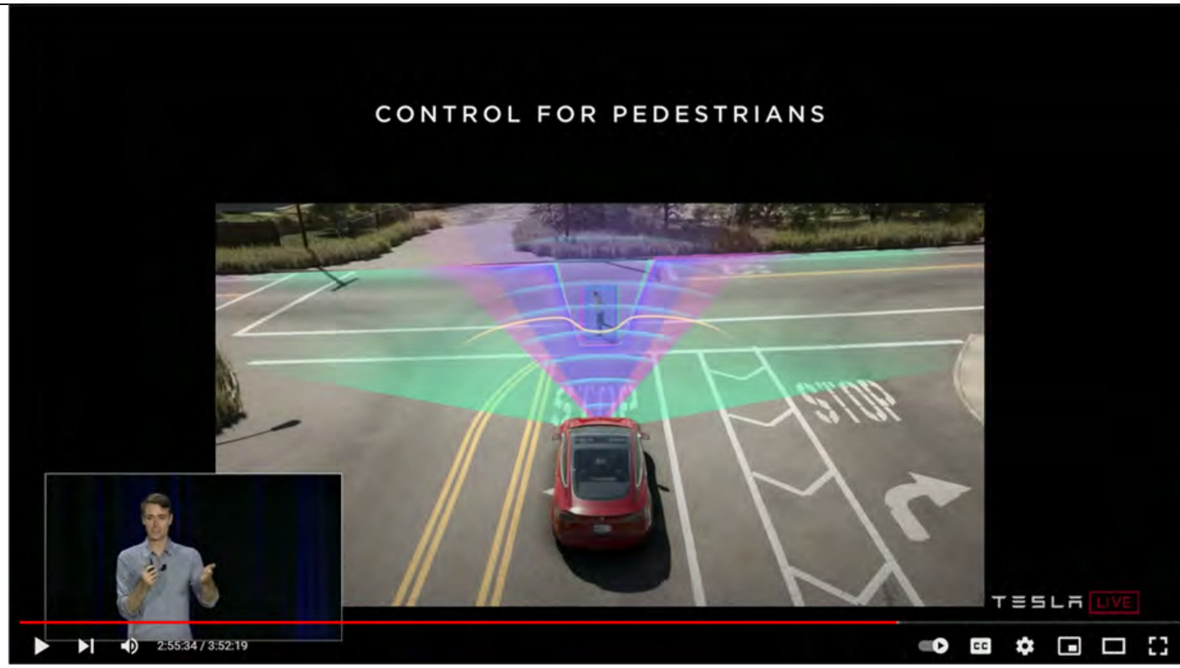
<table>
<tr><td></td><td>from the real world I'm just trying to imitate how people drive in real worlds and we can also apply the same data engine crank to all of this and make this work.”
https://www.youtube.com/watch?v=-b041NXGPZ8 at 1:04:10.

In this way, driving knowledge is: (i) learned on a first Tesla vehicle, (ii) transferred to the DoJo super computer where it is synthesized with other driving knowledge, (iii) packaged and sent from the DoJo to the fleet, e.g., the driving knowledge is stored on a memory of a second Tesla vehicle via the fleet over-the-air (OTA) software update, and (iv) accessed on the second Tesla vehicle for its autonomous driving. The second Tesla vehicle can, thereby, implement the driving knowledge learned on the first Tesla vehicle and synthesized on the DoJo.</td></tr>
<tr><td>[1a] one or more processors configured to perform at least:</td><td>Each of the accused Tesla vehicles (Models 3, S, X, Y, etc.) includes one or more processors (e.g., the full self-driving computer/chip) configured to perform full self-driving.

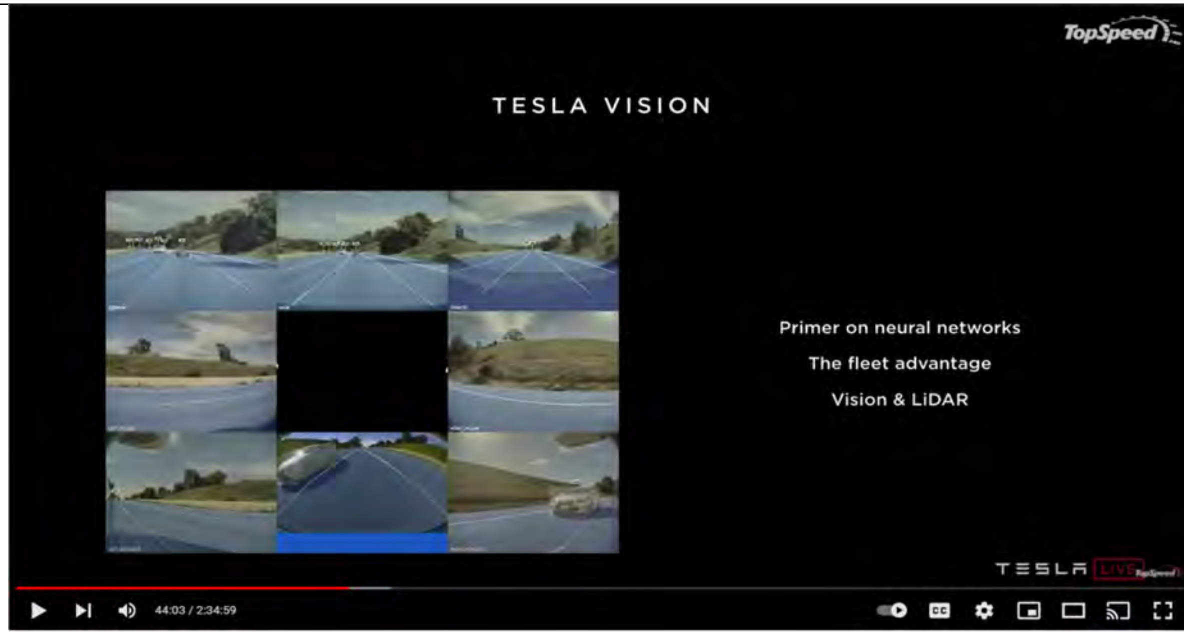
See Tesla Autonomy Day 2019 video https://www.youtube.com/watch?v=-b041NXGPZ8 at 7:11 (Tesla full self driving computer) and at 10:22 (Tesla full self driving chip).</td></tr>
<tr><td>[1b] accessing a memory that stores at least a knowledgebase that includes a first circumstance representation correlated with a first one or more instruction sets for operating a first device,</td><td>The processor of the second Tesla vehicle (the claimed “second device”) accesses its memory that stores at least a knowledgebase that includes a first circumstance representation (e.g., representation of pedestrians, other vehicles, roads, buildings, etc.) correlated with a set of driving instructions (e.g., driving instructions for effecting speed, steering, breaking, trajectory, etc.; the claimed “first one or more instruction sets”) for operating a first Tesla vehicle (the claimed “first device”).</td></tr>
</table>

wherein the first circumstance representation represents a first circumstance detected at least in part by one or more sensors of the first device, and wherein at least a portion of the first circumstance representation or at least a portion of the first one or more instruction sets for operating the first device is learned in a learning process that includes operating the first device at least partially by a user;	 See Tesla AI Day 2021 video https://www.youtube.com/watch?v=j0z4FweCy4M at 1:31:58 (example of the claimed "circumstance representation")

	 See Tesla AI Day 2021 video https://www.youtube.com/watch?v=j0z4FweCy4M at 1:32:40-48 (combining everything together we can produce these amazing data sets that annotate all of the road texture or the static objects and all of the moving objects **[example of the claimed "circumstance representation"]**)



See Tesla AI Day 2021 video https://www.youtube.com/watch?v=j0z4FweCy4M at 2:55:34 (discussing an automatic emergency breaking system). As shown in the clip, each Tesla vehicle has sensors such as cameras, etc. that are used to detect a first circumstance, e.g., a pedestrian in front of the vehicle. The first circumstance representation, therefore, represents a circumstance such as a person in front of the vehicle **(the claimed "wherein the first circumstance representation represents a first circumstance detected at least in part by one or more sensors of the first device").**

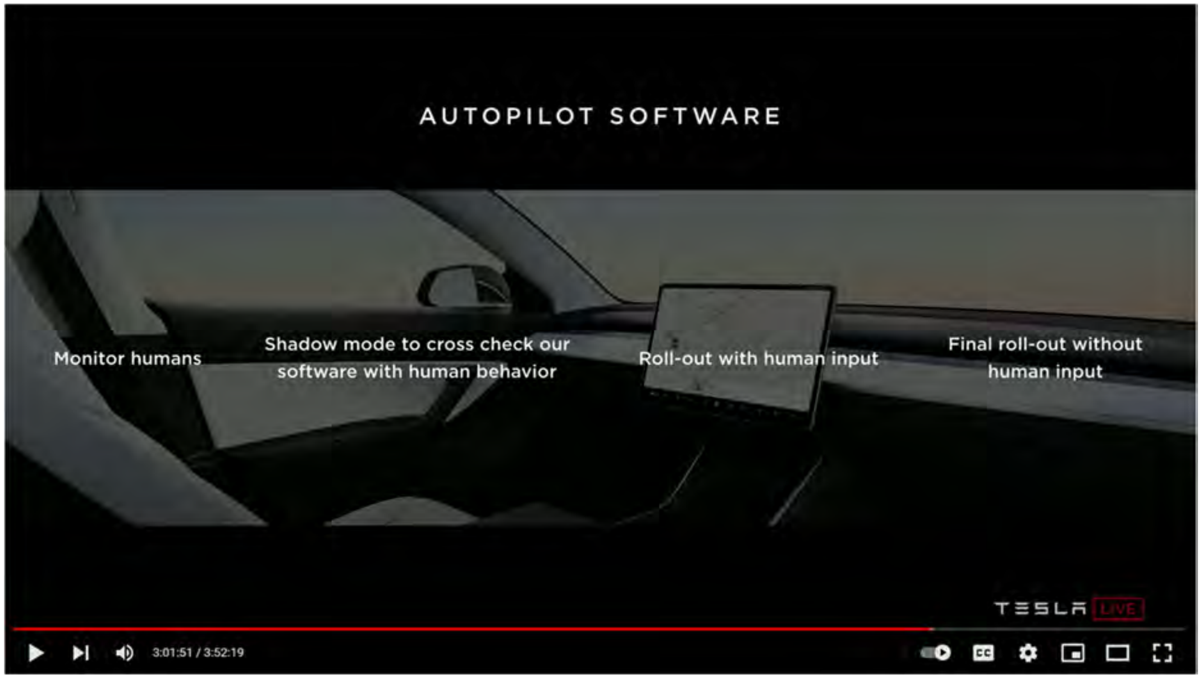


See also Tesla Autonomy Day 2019 video https://www.youtube.com/watch?v=-b041NXGPZ8 at 44:00 (a stream of videos from eight cameras **[the claimed "one or more sensors"]** across the vehicle used to make a lane change).

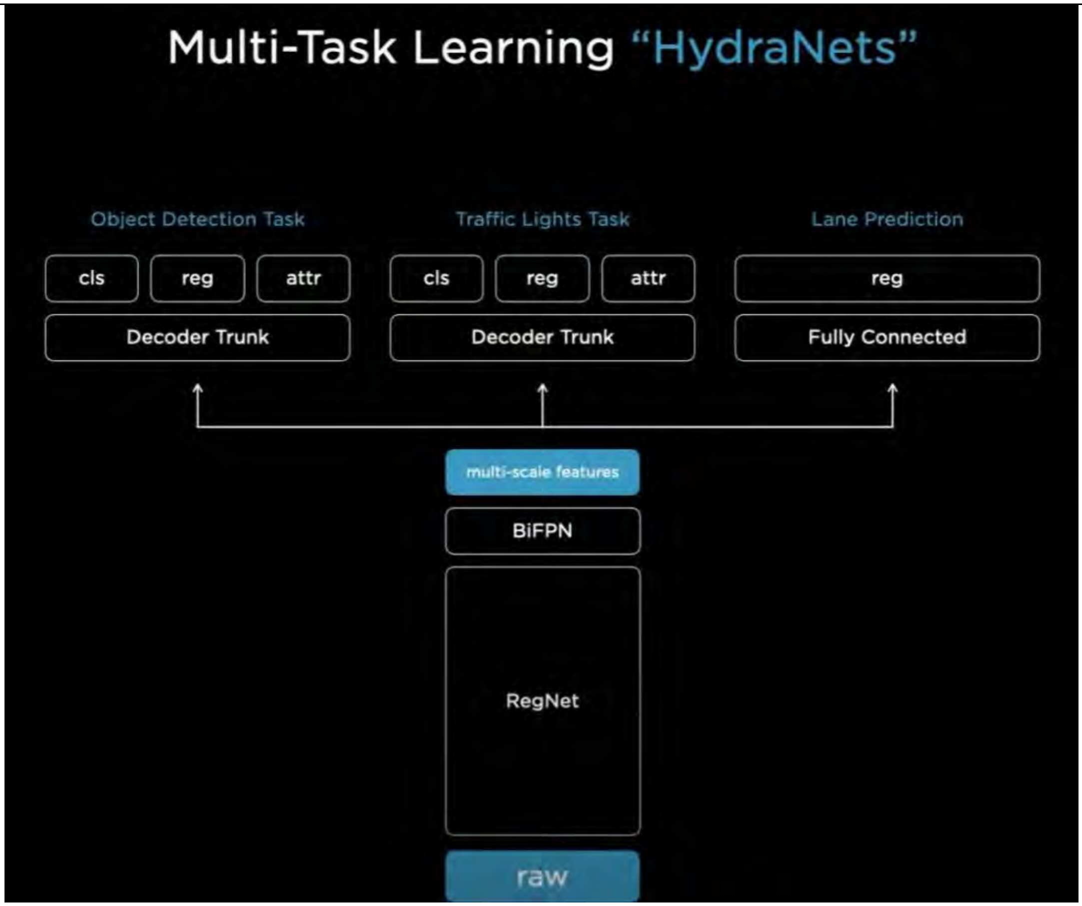
As detailed above, the first circumstance representation (e.g., a representation of a pedestrian in front the vehicle, representation of surrounding vehicles in a lane change situation, animals or debris in the road, a stop light, a stop sign, etc.) and the corresponding driving instructions (e.g., instructions for applying the breaks so the pedestrian is not hit or so that traffic laws are obeyed, or instructions for turning the wheel to safely change lanes or avoid an animal or debris, etc.) are learned in a process that involves a driver **(the claimed "user")** operating the first Tesla vehicle **(the claimed "wherein at least a portion of the first circumstance representation or at least a portion of the first one or more instruction sets for operating the first device is learned in a learning process that includes operating the first device at least partially by a user")**.

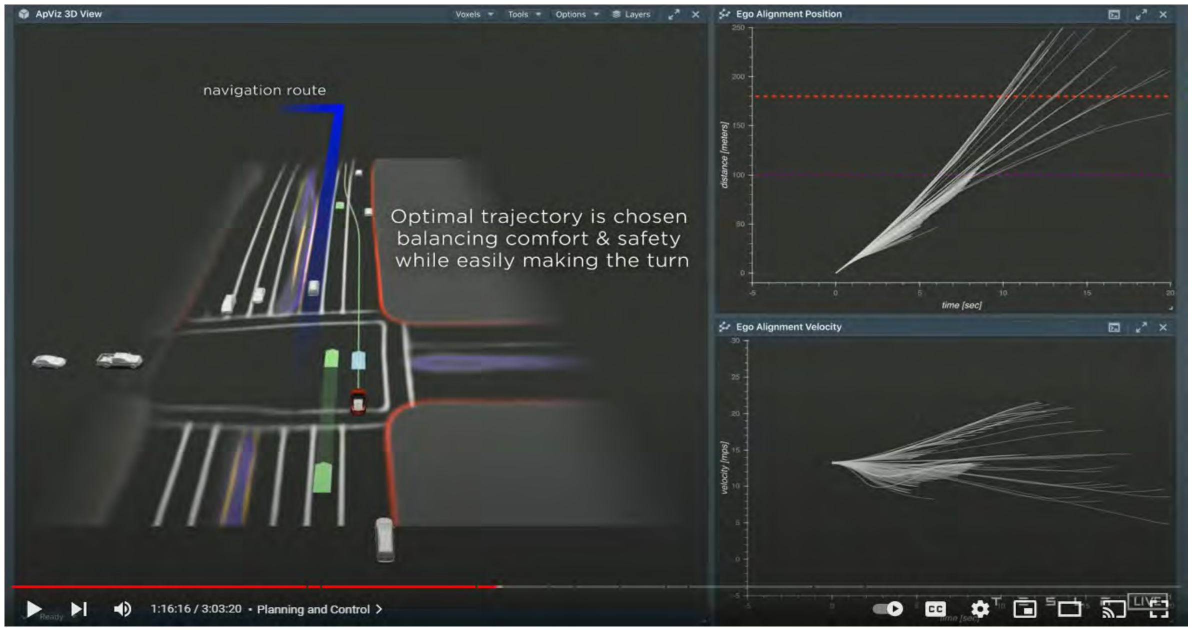
See Tesla Autonomy Day 2019 video https://www.youtube.com/watch?v=-b041NXGPZ8 at 1:04:10 (while you **[the claimed "user"]** are driving a car what you're actually doing is you are ***annotating the data because you are steering the wheel*** **[e.g., learned instruction]** you're telling us how to traverse different environments so what we're looking at here is some person in the fleet who took a left through an intersection and what we do here is we have the full video of all the cameras and we know that the path that this person took because of the GPS, the inertial measurement unit, the wheel angle, the wheel ticks, so we put all that together and we understand the path that this person took through this environment **[circumstance representation representing the environment correlated with driving instructions on how to traverse the environment]**. and then of course we can use this for supervision for the network so we just source a lot of this from the fleet, we train a neural network on those trajectories, and then the neural network predicts paths just from that data. … we're taking human trajectories from the real world we're just trying to imitate how people drive in real worlds)

See Tesla AI Day 2021 video https://www.youtube.com/watch?v=j0z4FweCy4M at 2:55:29 (all the human drivers are essentially training the neural net as to what is the correct course of action **[the claimed driving instructions]**)

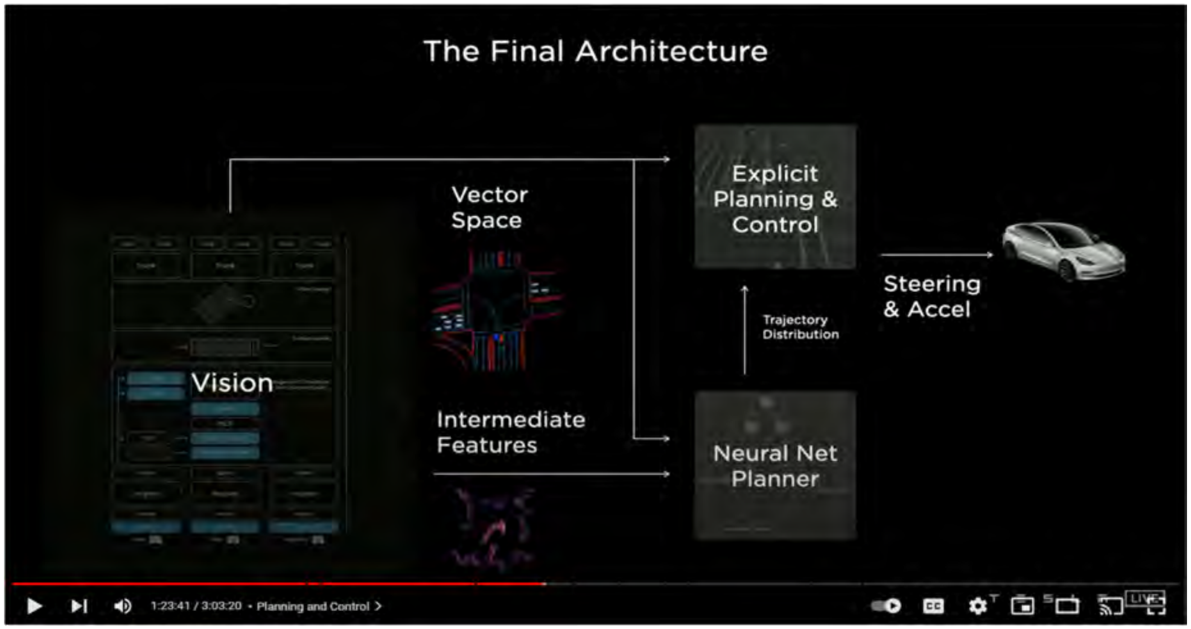
	 *See* Musk, Karpathy, etc. in Tesla Autonomy Day 2019 video at https://www.youtube.com/watch?v=-b041NXGPZ8 at 1:03:50 (everyone is training the network all the time), 1:30:55 (Tesla system learns from drivers training the system), and 1:52:23 (high level chart clearly showing monitoring and learning from human drivers and implementing autonomous driving based on the learned knowledge).
[1c] generating or receiving a second circumstance representation, wherein the second circumstance representation represents a second circumstance detected at least in part by: the one or more sensors of the first device, ***or one or***	The processor of the second Tesla vehicle **(the claimed "second device")** generates a **second circumstance representation** that represents a current circumstance (e.g., pedestrians, other vehicles, roads, buildings, animals, debris, etc.) detected at least in part by its cameras **(the claimed "one or more sensors of a second device")**. For instance, when the system is deployed to the fleet, a second Tesla vehicle **(the claimed "second device")** generates a **second circumstance representation** (e.g., representation of a different pedestrian in front the vehicle, representation of different surrounding vehicles in a lane change situation, representation of a different intersection in a left turn situation, etc.) that is detected by the second Tesla vehicle's sensors (e.g. cameras). The **second circumstance representation** is generated

more sensors of a second device;	so it can be compared with previously learned circumstance representations **(the claimed “first circumstance representation”)**, and if a similar match **(the claimed “at least partial match” in claim element 1d)** is found, the correlated driving instructions **(the claimed “first one or more instruction sets” in claim element 1b)** can be used for autonomous driving. Analogous evidence as for **“first circumstance representation”** and **“one or more sensors”** recited in claim element 1b applies to **“second circumstance representation”** and **“one or more sensors”** here (e.g., information detected by the second car’s sensors), and it is not duplicated to save space. See claim element 1b. As explained during the 2021 AI day, Tesla said it “discovered that we don’t just want to detect cars, we want to do a large number of tasks. So for example, we want to do traffic light recognition and detection, a lane prediction and so on. So very quickly, we conversioned this kind of architectural layout, where there’s a common shared backbone, and then branches off into a number of heads. So we call these therefore Hydra Nets. And these are the heads of the Hydra.” https://elon-musk-interviews.com/2021/08/31/tesla-ai-day-the-presentation-i/

	 https://elon-musk-interviews.com/2021/08/31/tesla-ai-day-the-presentation-i/
[1d] anticipating the first one or more instruction sets for operating the first device based on at least partial match between the second circumstance representation and the first circumstance	The processor of the second Tesla vehicle **(the claimed "second device")** anticipates a set of driving instructions (e.g., instructions for applying the breaks so the pedestrian is not hit, instructions for turning the wheel to safely change lanes or to avoid an animal or debris, instructions for turning left in an intersection, etc.; **the claimed "first one or more instruction sets"**) based on similarity **(the claimed "at least partial match")** between the **first circumstance representation** (e.g., a representation of a pedestrian in front the vehicle, a representation of surrounding vehicles in a lane change situation, a representation of an animal or debris, representation of an intersection in a left turn situation, etc. as previously learned on the first Tesla vehicle) and a **second circumstance**

representation; and	**representation** (e.g., a representation of a pedestrian in front the vehicle, a representation of surrounding vehicles in a lane change situation, a representation of an animal or debris, representation of an intersection in a left turn situation, etc. as currently detected by the second Tesla vehicle). Therefore, Tesla vehicles anticipate previously learned driving instructions based on similarity between the currently generated circumstance representation and previously learned circumstance representations.  See Tesla AI Day 2021 video https://www.youtube.com/watch?v=j0z4FweCy4M at 1:15:26 (let's see an example of how the search **[e.g., search for previously learned driving instructions for effecting a trajectory correlated with a previously learned circumstance representation that at least partially matches the incoming circumstance representation]** operates. so here we're trying to do a lane change. in this case the car needs to do two back to back lane changes to make the left turn up ahead. for this, the car searches over different maneuvers. so the first one it searches is a lane change that's close by but the car breaks pretty harshly so it's pretty uncomfortable. the next maneuver it tries does the lane change a bit late so it speeds up, goes beyond the other car, goes in front of the other cars, and find it at the lane change but now it risks missing the left turn. we do thousands of such searches in

	a very short time span). *See also id.* at 2:32:42 (similarly, for planning, we need to bake in a search **[e.g., search for previously learned driving instructions for effecting a trajectory correlated with a previously learned circumstance representation that at least partially matches the incoming circumstance representation]** and optimization into the planning, into the network architecture, and once we do that we should be able to do planning very quickly).
[1e] at least in response to the anticipating, executing the first one or more instruction sets for operating the first device, wherein the first device or the second device autonomously performs one or more operations defined by the first one or more instruction sets for operating the first device.	The processor of the second Tesla vehicle **(the claimed "second device")** executes, at least in response to the anticipating (claim element 1d), the previously learned set of driving instructions **(the claimed "first one or more instruction sets for operating the first device")** so that the second vehicle **(the claimed "second device")** can drive autonomously based on the driving instructions learned on the first Tesla vehicle **(the claimed "first device")**. For instance, memory of the second Tesla vehicle stores a **first circumstance representation** (e.g., representation of a pedestrian in front the vehicle, representation of surrounding vehicles in a lane change situation, representation of an intersection in a left turn situation, etc.) correlated with a set of driving instructions (e.g., instructions for applying the breaks so the pedestrian is not hit, instructions for turning the wheel to safely change lanes, instructions for turning left in an intersection, etc.; **the claimed "first one or more instruction sets"**) that have previously been learned on a first Tesla vehicle **(the claimed "first device")** and have been transferred to the second Tesla vehicle **(the claimed "second device")** via the fleet over-the-air (OTA) software update. In response to determining that the incoming circumstance representation (e.g., representation of a pedestrian currently in front the vehicle, representation of currently surrounding vehicles in a lane change situation, representation of a current intersection in a left turn situation, etc.) is similar to **(the claimed "at least partial match")** the previously learned circumstance representation (e.g., representation of a pedestrian previously in front the vehicle, representation of previously surrounding vehicles in a lane change situation, representation of a previous intersection in a left turn situation, etc.), the processor of the second Tesla vehicle **(the claimed "second device")** causes the previously learned driving instruction (e.g., instructions for applying the breaks so the pedestrian is not hit, instructions for turning the wheel to safely change lanes, instructions for turning left in an intersection, etc.) to be executed. As stated by Elon Musk on Tesla Autonomy Day 2019, Tesla system distributes driving knowledge learned from multiple Tesla vehicles to all Tesla vehicles via over-the-air (OTA) software updates, thereby enabling the second Tesla vehicle **(the claimed "second device")** to autonomously implement driving knowledge learned on the first vehicle **(the claimed "first device")** as claimed. Mr. Musk and

	Mr. Karpathy describe this as a crucial competitive advantage of Tesla over other autonomous driving companies. *See* Musk, Karpathy, etc. in Tesla Autonomy Day 2019 video https://www.youtube.com/watch?v=-b041NXGPZ8 at 38:58 ("I think a very powerful sustainable advantage for us is the fleet"), 44:40 ("it is such a big deal that we have the fleet"), 55:55 ("why is Tesla in such a unique and interesting position to really get all these three essentials right, and the answer to that of course is the fleet"), 122:14 ("it's extremely difficult to catch up when Tesla has 100 times more miles per day than everyone else combined").  See Tesla AI Day 2021 video https://www.youtube.com/watch?v=j0z4FweCy4M at 1:23:40 (so this is what a final architecture is going to look like. the vision system is going to crush down the dense video data into a vector space **[incoming circumstance representation representing objects in the second Tesla vehicle's surrounding]**. it's going to be consumed by both an explicit planner and a neural network planner **[CNN that stores previously learned circumstance representations correlated with previously learned driving instructions and that searches for at least partially matching circumstance representation and correlated driving instructions]** in addition to this, the network planner can also consume intermediate features of the network. Together, this produces a trajectory

	distribution and it can be optimized end to end both with explicit cost functions and human intervention and other imitation data. this then goes into explicit planning function that does whatever it sees for that and produces the final steering and acceleration commands for the car).
Claim 3	**Exemplary Infringement Evidence**
3. The system of claim 1, wherein the first circumstance representation includes a first one or more object representations, and wherein the second circumstance representation includes a second one or more object representations.	Each autonomous Tesla vehicle with Software Version 9.0 and beyond (this includes vehicles with enhanced autopilot and/or full self-driving (FSD)) alone or together with the Dojo super computer includes the system of claim 1. Also, the first circumstance representation includes a first one or more object representations, and wherein the second circumstance representation includes a second one or more object representations. For example, the processor of the Tesla vehicle accesses its memory that stores at least a knowledgebase that includes a first circumstance representation including one or more object representations (e.g., representation of pedestrians, other vehicles, roads, buildings, etc.). And the second circumstance representation received or generated by the Tesla vehicle includes a second one or more object representations (e.g., representation of pedestrians, other vehicles, roads, buildings, etc.).

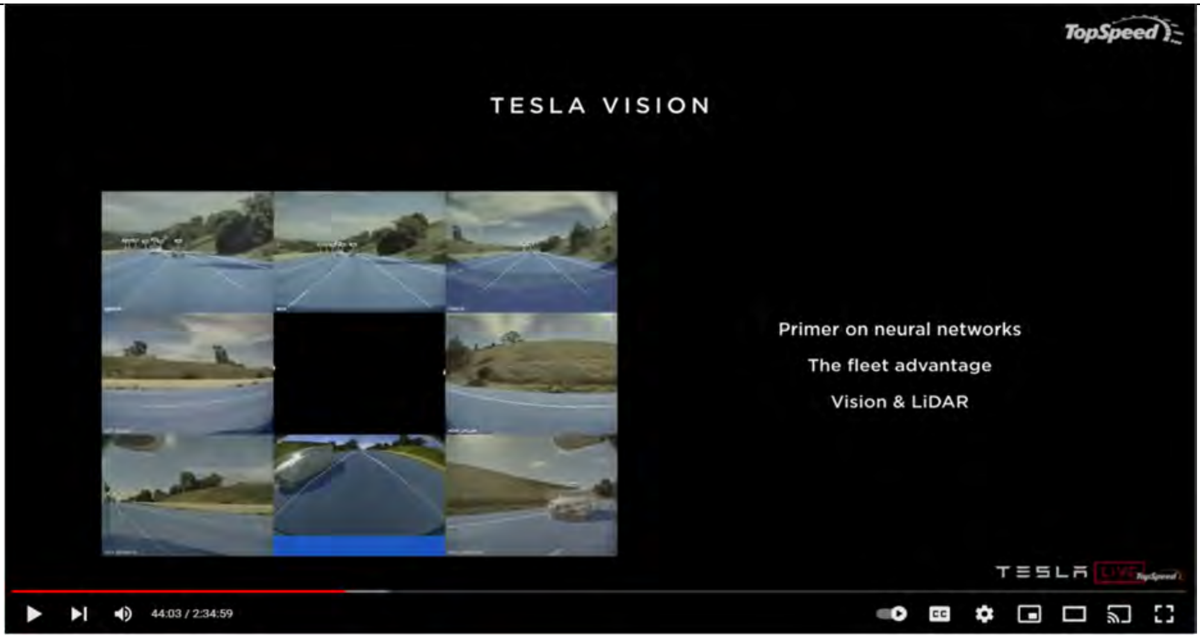
	 See Tesla AI Day 2021 video https://www.youtube.com/watch?v=j0z4FweCy4M at 1:31:58 (example of the claimed "circumstance representations" with multiple object representations).

Exhibit J

U.S. Patent No. 11,055,583	
Claim 4	**Exemplary Infringement Evidence**[1]
[1pre] A system comprising:	To the extent the preamble is limiting, each autonomous Tesla vehicle with Software Version 9.0 and beyond (this includes vehicles with enhanced autopilot and/or full self-driving (FSD)) alone or together with the Dojo super computer meet the limitations of the claimed system. As a driver drives a Tesla vehicle, the vehicle learns digital pictures depicting the vehicle's surrounding (i.e. pedestrians, other vehicles, roads, buildings, etc.) as well as driving instructions (i.e. instructions for effecting speed, steering, breaking, trajectory, etc.) that the driver used to navigate that surrounding. This driving knowledge learned on one Tesla vehicle is then distributed to all vehicles in Tesla fleet via over-the-air (OTA) software updates. The fleet, therefore, enables each vehicle in the fleet to autonomously implement driving instructions learned on one or more originating vehicles when digital pictures of a similar surrounding are captured by any of the vehicles in Tesla fleet, thereby enabling their autonomous driving. The discussion and evidence cited in claims [1a-d] are incorporated herein.
[1a] one or more processors; and one or more non-transitory machine readable media storing machine readable code that, when executed by the one or more processors, causes the one or more processors to perform at least:	Each of the accused Tesla vehicles (Models 3, S, X, Y, etc.) includes one or more processors (e.g., the full self-driving chip) programmed by code stored on one or more non-transitory machine readable media (i.e. RAM memory, SSD drive, flash memory, hard drive, etc.) all part of Tesla full self-driving computer.   *See* Tesla Autonomy Day 2019 video https://www.youtube.com/watch?v=-b041NXGPZ8 at 7:11 (Tesla

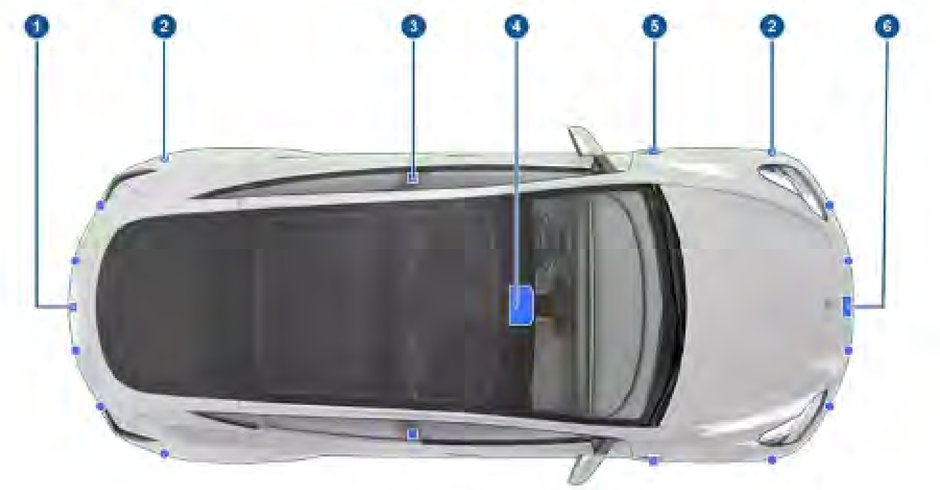
[1] These infringement contentions are prepared with publicly available information.

	full self driving computer) and at 10:22 (Tesla full self driving chip). **How does Autopilot work?** As of mid-February 2022, all vehicles built for the North American market will feature Tesla Vision, which uses eight cameras and powerful neural net processing to see the environment around the car and deliver Autopilot features. This camera suite provides occupants with an awareness of their surroundings that a driver alone would not otherwise have. A powerful onboard computer processes these inputs in a matter of milliseconds to help make driving safer and less stressful. *See* https://www.tesla.com/support/autopilot
[1b] receiving or generating a first one or more digital pictures, wherein the first one or more digital pictures depict at least a portion of a first device's surrounding;	Each autonomous Tesla vehicle is an example of a system including one or more processors that receiving or generating a first one or more digital pictures, wherein the first one or more digital pictures depict at least a portion of a first device's surrounding. For example, as a driver drives a first Tesla vehicle **(the claimed "first device")**, the processor of the first Tesla vehicle receives from the vehicle's cameras the **first one or more digital pictures** depicting the vehicle's surrounding (i.e. pedestrians, other vehicles, roads, buildings, etc.).

	 *See* Tesla Autonomy Day 2019 video https://www.youtube.com/watch?v=-b041NXGPZ8 at 44:00 (a stream of videos from eight cameras across the vehicle used to make a lane change). Further, the Model Y (an accused vehicle) has multiple cameras to depict a portion of the vehicle's surrounding.

How It Works

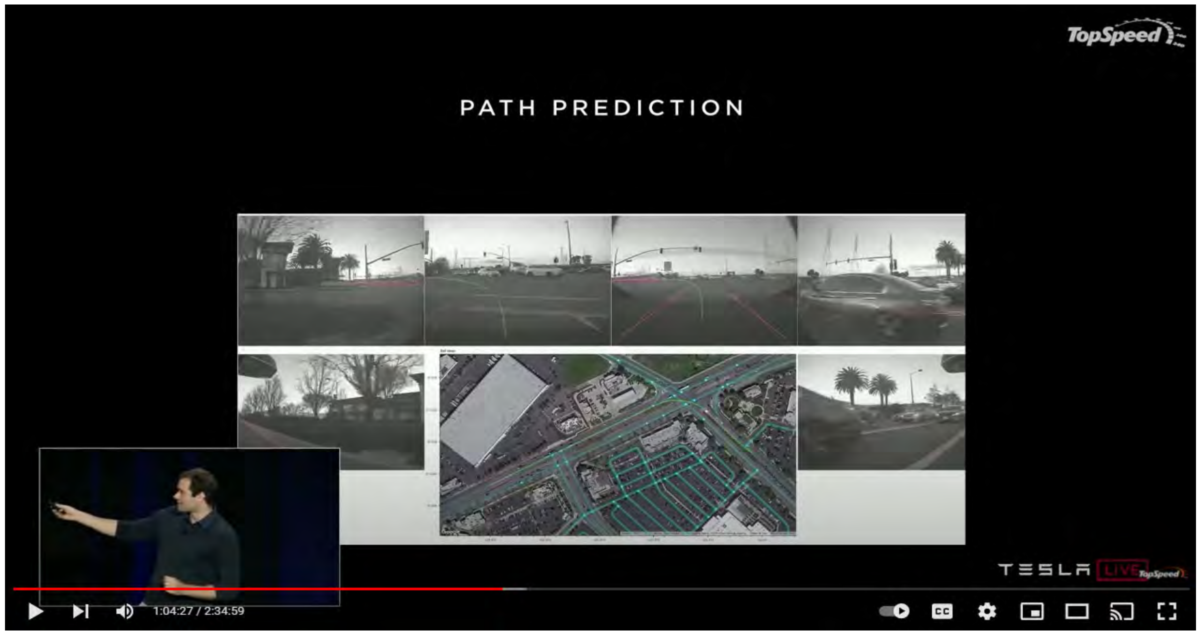
Your Model Y includes the following components that actively monitor the surrounding area:



1. A camera is mounted above the rear license plate.
2. Ultrasonic sensors (if equipped) are located in the front and rear bumpers.
3. A camera is mounted in each door pillar.
4. Three cameras are mounted to the windshield above the rear view mirror.
5. A camera is mounted to each front fender.
6. Radar (if equipped) is mounted behind the front bumper.

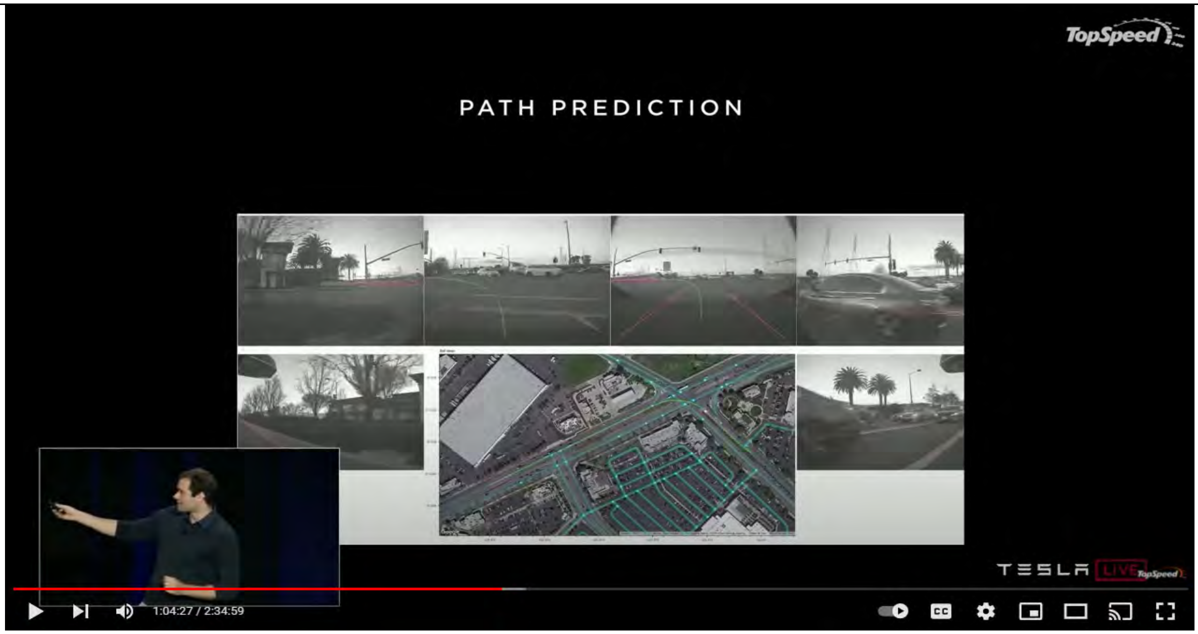
Model Y is also equipped with high precision electronically-assisted braking and steering systems.

See https://www.tesla.com/ownersmanual/modely/en_us/GUID-EDA77281-42DC-4618-98A9-CC62378E0EC2.html

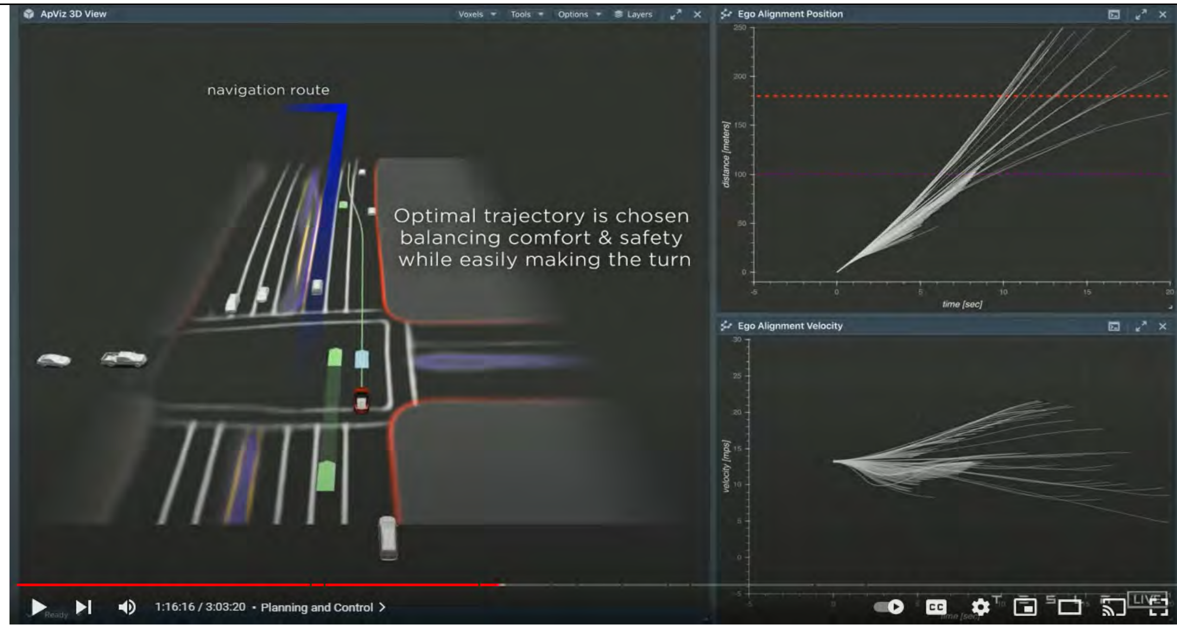
[1c] receiving or generating a first one or more instruction sets for operating the first device; and	Each autonomous Tesla vehicle is an example of a system including one or more processors can receiv[e] or generat[e] a first one or more instruction sets for operating the first device. For example, as the driver drives the first Tesla vehicle **(the claimed "first device")**, the processor of the first Tesla vehicle receives a set of driving instructions (i.e. driving instructions for effecting speed, steering, breaking, trajectory, etc.; **the claimed "first one or more instruction sets"**) that the driver used to navigate the vehicle's surrounding.  *See* Tesla Autonomy Day 2019 video https://www.youtube.com/watch?v=-b041NXGPZ8 at 1:04:10 ("While you are driving a car [**the claimed "first device"**] what you're actually doing is you are annotating the data because you are steering the wheel. You're telling us how to traverse different environments so what we're looking at here is some person in the fleet who took a left through an intersection and what we do here is we have the full video of all the cameras and we know that the path that this person took because of the GPS, the inertial measurement unit, the wheel angle, the wheel ticks, so we put all that together and we understand the path that this person took through this environment **[the claimed receiving of**

	instructions]. And then of course we can use this for supervision for the network so we just source a lot of this from the fleet, we train a neural network on those trajectories, and then the neural network predicts paths just from that data. … we're taking human trajectories from the real world we're just trying to imitate how people drive in real worlds.") *See also* Tesla AI Day 2021 video https://www.youtube.com/watch?v=j0z4FweCy4M at 2:55:29 (all the human drivers are essentially training the neural net as to what is the correct course of action **[the claimed driving instructions]**)
[1d] learning the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device.	Each autonomous Tesla vehicle is an example of a system including one or more processors can learn[] the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device. For example, as the driver drives the first Tesla vehicle **(the claimed "first device")**, the processor of the first Tesla vehicle learns (including storing what is learned in memory) the first one or more digital pictures depicting the vehicle's surrounding (i.e. pedestrian in front the vehicle, surrounding vehicles in a lane change situation, intersection in a left turn situation, etc.) and a set of driving instructions (i.e. instructions for applying the brakes so the pedestrian is not hit, instructions for turning the wheel to safely change lanes, instructions for turning left in an intersection, etc.) that the driver used to navigate the vehicle's surrounding **(the claimed "learning the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device")**.

	 *See* Tesla Autonomy Day 2019 video https://www.youtube.com/watch?v=-b041NXGPZ8 at 1:03:50 (Musk: "Everyone is training the network all the time"), 1:30:55; 1:52:23.

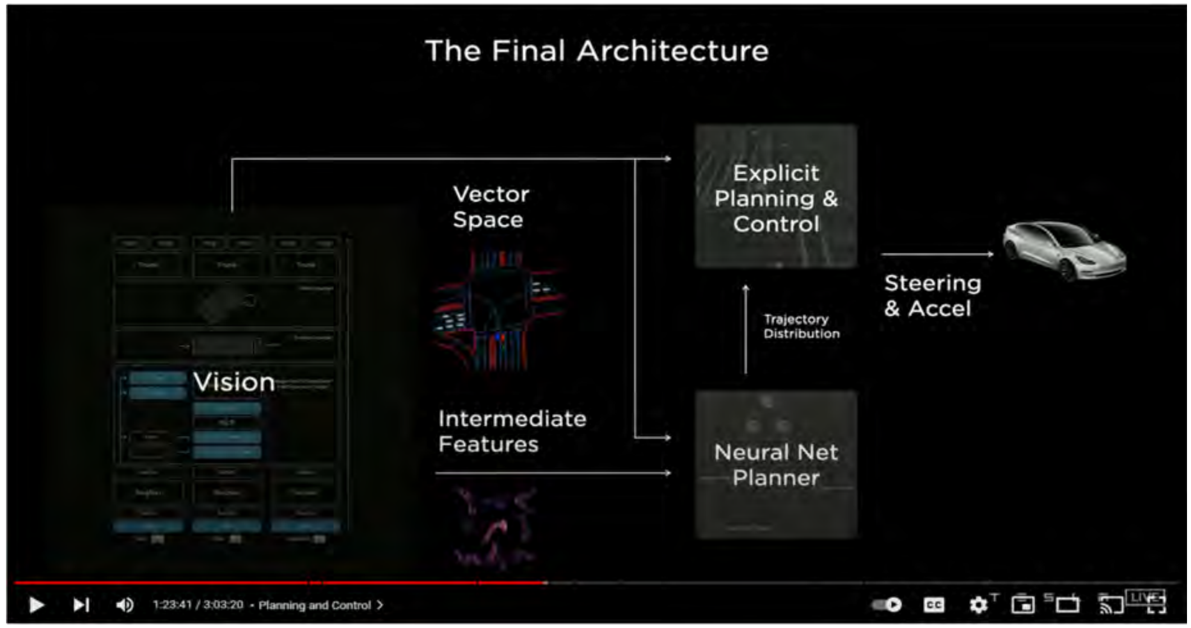
	 See Tesla Autonomy Day 2019 video https://www.youtube.com/watch?v=-b041NXGPZ8 at 1:04:10 ("While you are driving a car **[the claimed "first device"]** what you're actually doing is you are annotating the data because you are steering the wheel **[the claimed "learning the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device"]** you're telling us how to traverse different environments so what we're looking at here is some person in the fleet who took a left through an intersection and what we do here is we have the full video of all the cameras and we know that the path that this person took because of the GPS, the inertial measurement unit, the wheel angle, the wheel ticks, so we put all that together and we understand the path that this person took through this environment. And then of course we can use this for supervision for the network so we just source a lot of this from the fleet, we train a neural network on those trajectories, and then the neural network predicts paths just from that data. … we're taking human trajectories from the real world we're just trying to imitate how people drive in real worlds").
[4pre] the system of claim 1, wherein the machine readable	Each autonomous Tesla vehicle is an example of a system that includes a processor, memory, etc. Tesla fleet includes over a million of such customer vehicles. A fleet of Tesla vehicles is also an example of a covered system.

code, when executed by the one or more processors, causes the one or more processors to further perform at least:	The discussion and evidence cited in claims [1a-d] and [4a-c] are incorporated herein
[4a] receiving or generating a new one or more digital pictures;	Each autonomous Tesla vehicle is an example of a system including processors that can receive or generate a new one or more digital pictures. For example, the processor of the second Tesla vehicle **(the claimed "second device")** receives from the vehicle's cameras the **new one or more digital pictures** depicting the vehicle's current surrounding (i.e. pedestrians, other vehicles, roads, buildings, etc.). The discussion and evidence cited in claims [1b] is incorporated herein.
[4b] determining the first one or more instruction sets for operating the first device based on at least partial match between the new one or more digital pictures and the first one or more digital pictures; and	Each autonomous Tesla vehicle is an example of a system including processors that can determine the first one or more instruction sets for operating the first device based on at least partial match between the new one or more digital pictures and the first one or more digital pictures. For example, the processor of the second Tesla vehicle **(the claimed "second device")** determines a set of driving instructions (i.e. instructions for applying the breaks so the pedestrian is not hit, instructions for turning the wheel to safely change lanes, instructions for turning left in an intersection, etc.; **the claimed "first one or more instruction sets"**) based on similarity **(the claimed "at least partial match")** between the **first one or more digital pictures** depicting the first vehicle's surrounding (i.e. pedestrian in front the vehicle, surrounding vehicles in a lane change situation, intersection in a left turn situation, etc. as previously learned on the first Tesla vehicle) and the **new one or more digital pictures** depicting the second vehicle's surrounding (i.e. different pedestrian in front the vehicle, different surrounding vehicles in a lane change situation, different intersection in a left turn situation, etc. as currently captured by the cameras of the second Tesla vehicle). Therefore, Tesla vehicles determine previously learned driving instructions based on similarity between the currently received digital pictures and previously learned digital pictures.

	 See Tesla AI Day 2021 video https://www.youtube.com/watch?v=j0z4FweCy4M at 1:15:26 ("Let's see an example of how the search **[i.e. search for previously learned driving instructions for effecting a trajectory correlated with previously learned digital pictures that at least partially match the incoming digital pictures]** operates. So here we're trying to do a lane change. In this case the car needs to do two back to back lane changes to make the left turn up ahead. For this, the car searches over different maneuvers. So the first one it searches is a lane change that's close by but the car breaks pretty harshly so it's pretty uncomfortable. The next maneuver it tries does the lane change a bit late so it speeds up, goes beyond the other car, goes in front of the other cars, and find it at the lane change but now it risks missing the left turn. We do thousands of such searches in a very short time span."). *See also id.* at 2:32:42 ("Similarly, for planning, we need to bake in a search **[i.e. search for previously learned driving instructions for effecting a trajectory correlated with previously learned digital pictures that at least partially match the incoming digital pictures]** and optimization into the planning, into the network architecture, and once we do that we should be able to do planning very quickly").
[4c] at least in	Each autonomous Tesla vehicle is an example of a system including processors that can at least in response

response to the determining, causing the first device or a second device to perform one or more operations defined by the first one or more instruction sets for operating the first device.	to the previous determining, cause the first device or a second device to perform one or more operations defined by the first one or more instruction sets for operating the first device. For example, the processor of the second Tesla vehicle **(the claimed "second device")** executes, at least in response to the determining (claim element 4b), the previously learned set of driving instructions for operating the first Tesla vehicle **(the claimed "first one or more instruction sets for operating the first device")** so that the second vehicle **(the claimed "second device")** can drive autonomously based on the driving instructions learned on the first Tesla vehicle **(the claimed "first device")**. Further, memory of the second Tesla vehicle stores the **first one or more digital pictures** depicting the first vehicle's surrounding (i.e. pedestrian in front the vehicle, surrounding vehicles in a lane change situation, intersection in a left turn situation, etc.) correlated with a set of driving instructions (i.e. instructions for applying the brakes so the pedestrian is not hit, instructions for turning the wheel to safely change lanes, instructions for turning left in an intersection, etc.; **the claimed "first one or more instruction sets"**) that have previously been learned on the first Tesla vehicle **(the claimed "first device")** and have been transferred to the second Tesla vehicle **(the claimed "second device")** via the fleet over-the-air (OTA) software update. In response to determining that the incoming **new one or more digital pictures** are similar to **(the claimed "at least partial match")** the previously learned **first one or more digital pictures**, the processor of the second Tesla vehicle causes the previously learned driving instruction (i.e. instructions for applying the brakes so the pedestrian is not hit, instructions for turning the wheel to safely change lanes, instructions for turning left in an intersection, etc.; **the claimed "first one or more instruction sets"**) to be executed to cause the second Tesla vehicle **(the claimed "second device")** to perform autonomous driving. As stated by Mr. Musk on Tesla Autonomy Day 2019, Tesla system distributes driving knowledge learned from multiple Tesla vehicles to all Tesla vehicles via over-the-air (OTA) software updates, thereby enabling the second Tesla vehicle **(the claimed "second device")** to autonomously implement driving knowledge learned on the first vehicle **(the claimed "first device")** as claimed. Mr. Musk and Mr. Karpathy describe this as a crucial competitive advantage of Tesla over other autonomous driving companies. *See* Tesla Autonomy Day 2019 video https://www.youtube.com/watch?v=-b041NXGPZ8 at 38:58 (Musk: "I think a very powerful sustainable advantage for us is the fleet"), 44:40 (Karpathy: "it is such a big deal that we have the fleet. . . why it [the fleet] is a key enabling factor"), 55:55 (Karpathy: "why is Tesla in such a unique and interesting position to really get all these three essentials right, and the answer

to that of course is the fleet"), 122:14 (Musk: "it's extremely difficult to catch up when Tesla has 100 times more miles per day than everyone else combined").



See Tesla AI Day 2021 video https://www.youtube.com/watch?v=j0z4FweCy4M at 1:23:40 ("So this is what a final architecture is going to look like. The vision system is going to crush down the dense video data into a vector space **[incoming digital pictures depicting the second Tesla vehicle's surrounding]**. It's going to be consumed by both an explicit planner and a neural network planner **[CNN that stores previously learned digital pictures correlated with previously learned driving instructions and that searches for at least partially matching digital pictures and correlated driving instructions]**. In addition to this, the network planner can also consume intermediate features of the network. Together, this produces a trajectory distribution and it can be optimized end to end both with explicit cost functions and human intervention and other imitation data. this then goes into explicit planning function that does whatever it sees for that and produces the final steering and acceleration commands for the car").

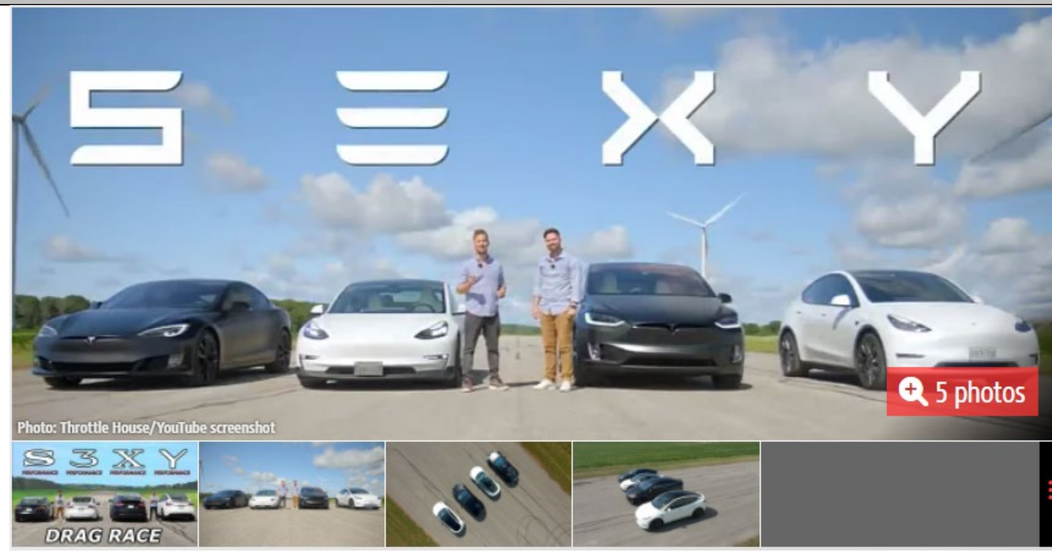
Exhibit K

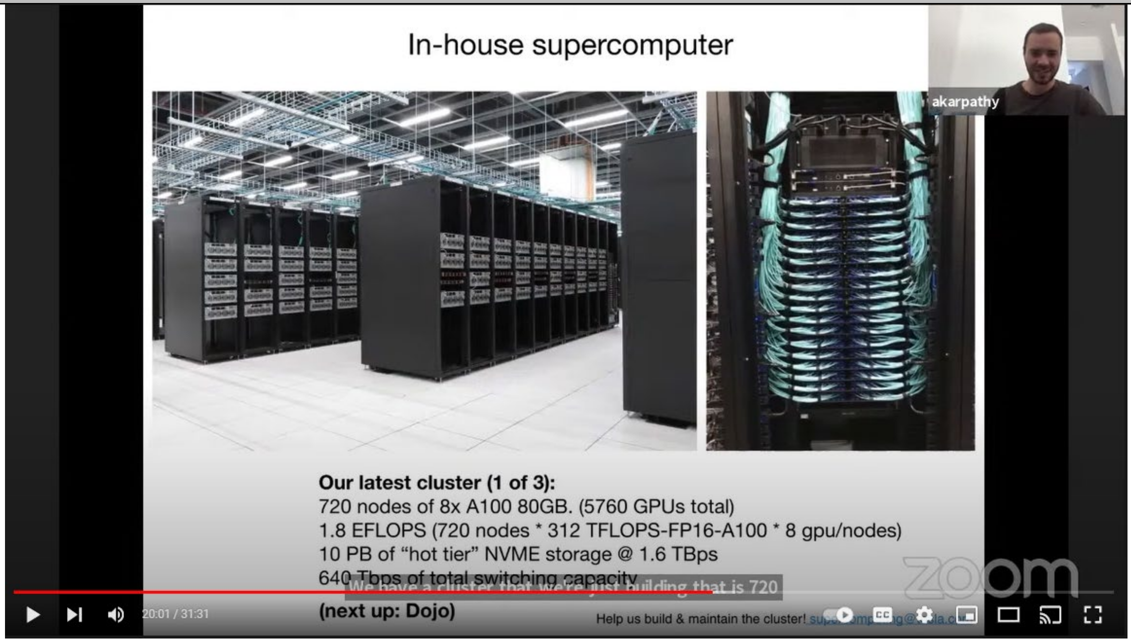
EXHIBIT K
AUTONOMOUS DEVICES
INFRINGEMENT CONTENTION CHART FOR U.S. PATENT NO. 11,663,474
CHARTED AGAINST TESLA'S MODELS S, 3, X, AND Y

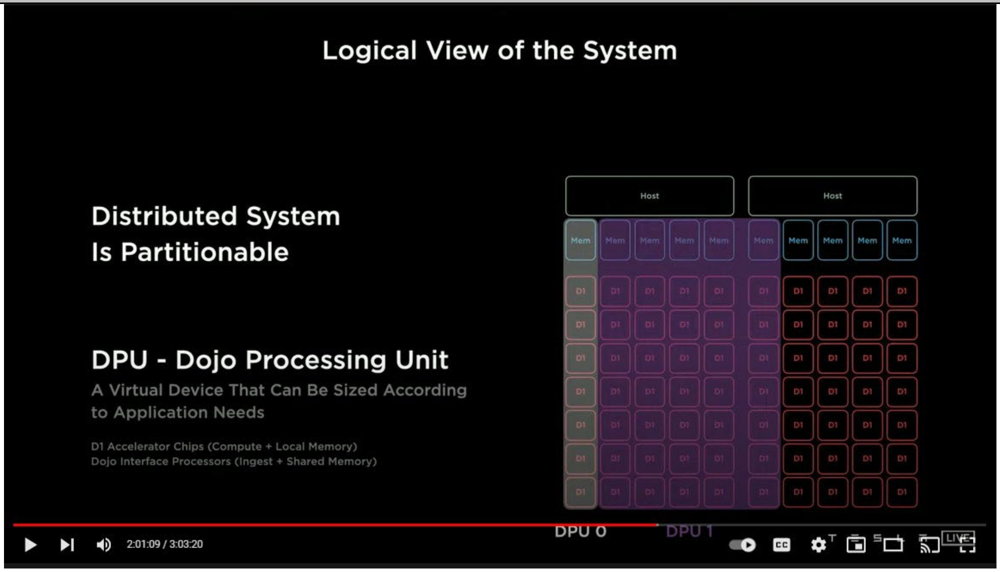
On August 15, 2023, the Court ordered Tesla to produce its source code and source code documents (collectively "SC files") in their native format. Throughout this litigation and including at the time of service of these preliminary infringement contentions, Tesla produced the SC files in adobe pdf, which the Court found to be "inefficient and cumbersome" and incompatible with "traditional source code review tools and capabilities." Despite this finding, Tesla still has not produced the SC files in their native format. Tesla also prevented AD lawyers and experts from collaboratively reviewing the SC files, forced AD's lawyers to destroy their working copies of the SC files and insisted on a byzantine framework for even acquiring the SC files. Despite the Court's ruling that AD's lawyers can have multiple dcopies of the SC files—Tesla has not yet complied with its obligations. As a result, AD reserves the right to modify this preliminary chart after Tesla complies with its obligations and AD's experts have an opportunity to review and process Tesla's production. Tesla also refused to provide complete responses to AD's interrogatory requests. AD reserves the right to modify this preliminary chart based on Tesla's eventual compliance with its discovery obligations.

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
[45pre] A first device comprising:	The '474 Accused Products are devices comprising the limitations below. For example, the 474 Accused Products include individual Tesla vehicles (e.g., Tesla's Models S, 3, X, and Y) as well as a supercomputer/simulator implements the following limitations.

[1] To the extent that any of the limitations of the asserted claims are not literally infringed as set forth in these contentions, Autonomous Devices reserves the right to amend these disclosures to include a theory under the doctrine of equivalents prior to the final contentions.
[2] The Accused Products for the '474 Patent (herein the "'474 Accused Products" or "the 474 Accused Products") comprise at least the Tesla Models S, 3, X, and Y (e.g., individual accused Tesla vehicles or a fleet thereof) executing Software Version 9.0 and beyond (including vehicles with enhanced autopilot and/or full self-driving (FSD)) alone or together with Tesla's simulator technology.

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 Source: https://www.autoevolution.com/news/tesla-model-s-vs-3-vs-x-vs-y-the-s3xy-performance-drag-race-is-here-147100.html Further, the 474 Accused Products are trained by Tesla using a supercomputer, e.g., the NVIDIA supercomputer and/or the DOJO supercomputer, which simulates driving behavior and issues instructions to control vehicles included in Tesla's fleet. *See, e.g.*, [CVPR'21 WAD] Keynote - Andrej Karpathy, Tesla, available at https://www.youtube.com/watch?v=g6bOwQdCJrc.

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 *See* Tesla AI Day video, available at https://www.youtube.com/watch?v=j0z4FweCy4M&t=3s > *"We do have a major program at Tesla which we don't have enough time to talk about today called "Dojo". That's a super powerful training computer. The goal of Dojo will be to be able to take in vast amounts of data and train at a video level and do unsupervised massive training of vast amounts of video with the Dojo program – or Dojo computer."* Elon Musk hints at Tesla's not-so-secret Dojo AI-training supercomputer capacity – Electrek

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	Karpathy commented on the effort: > *"We have a neural net architecture network and we have a data set, a 1.5 petabytes data set that requires a huge amount of computing. So I wanted to give a plug to this insane supercomputer that we are building and using now. For us, computer vision is the bread and butter of what we do and what enables Autopilot. And for that to work really well, we need to master the data from the fleet, and train massive neural nets and experiment a lot. So we invested a lot into the compute. In this case, we have a cluster that we built with 720 nodes of 8x A100 of the 80GB version. So this is a massive supercomputer. I actually think that in terms of flops, it's roughly the number 5 supercomputer in the world."* https://www.inputmag.com/tech/tesla-showed-off-its-massive-supercomputer-for-self-driving- data-processing
[45a]. a knowledgebase that includes one or more inputs for inputting at least a portion of a circumstance representation, wherein the one or more inputs are correlated with one or more instruction sets for operating a second device;	The '474 Accused Products are devices further comprising a knowledgebase that includes one or more inputs for inputting at least a portion of a circumstance representation, wherein the one or more inputs are correlated with one or more instruction sets for operating a second device. For example, the 474 Accused Products include a knowledgebase based on circumstance representations (or circumstances). To form/generate circumstances, Tesla vehicles, including those vehicles with enhanced autopilot and/or full self-driving (FSD), alone or in combination with the supercomputer/simulator, are equipped with a multitude of sensors (*see*, [45b] below for more info about the sensors included in Tesla vehicles). These sensors capture data which is then uploaded as an input to the simulator, e.g., via an internet uplink. Once uploaded/input to the simulator, the circumstances re correlated with instruction sets which are shared with the fleet, for example, also via an internet connection. At a high level, an example of a circumstance representation may be making a left-hand turn at a particular intersection. One or more object representations may be included within a circumstance representation. Examples of object representations that may be present within a circumstance representation.

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	Multiple circumstance representations, each including multiple object representations, may be received or generated by each vehicle included in the Tesla fleet (e.g., recorded by the cameras and/or sensed by ultrasonic/telemetry sensors). These circumstance representations are then added (e.g., uploaded) to the knowledgebase (e.g., the supercomputer/simulator and/or the trained CNN on the Tesla vehilce). The knowledge base can also include the FSD computer installed on the Tesla vehicles.  At/around 1:31:58 in Tesla's AI Day 2021 presentation depicts a circumstance representation (e.g., navigating through a particular intersection) with multiple object representations.

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	To further explain the relationship between circumstance representations/circumstances and object representations – Circumstance representations may be learned while drivers drive the accused Tesla vehicles. The '474 Accused Products also learn driving instructions, such as instructions for effecting the speed, steering, and/or braking of the accused Tesla vehicle being driven based on the actions taken by the driver to navigate those circumstances. This driving knowledge learned by one Tesla vehicle is transmitted to Tesla's simulator before being distributed to all vehicles in the Tesla fleet via over-the-air (OTA) software updates. Therefore, the fleet enables each accused Tesla vehicle included therein to autonomously implement driving instructions learned on one or more originating vehicles when similar circumstances are detected by any of the vehicles in Tesla's fleet, thereby enabling Tesla's autonomous driving capabilities. Further, to effectuate the circumstance representations, a Tesla vehicle detects objects. For example, in the previous analysis of driving behind the car with the bike mounted to it, the car detects the bike as an object. Tesla explains object detection as follows:

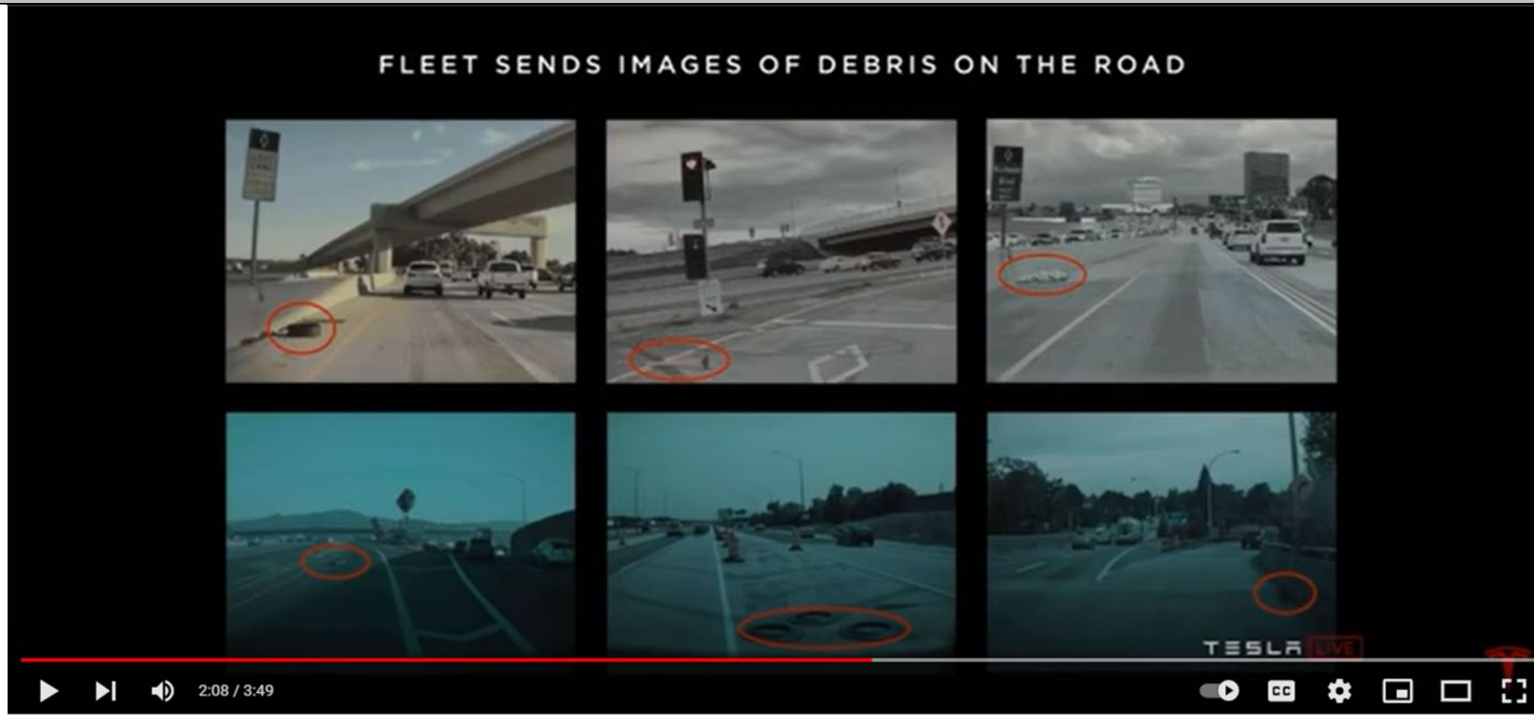
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 "so object detection is something we care a lot about we'd like to put bounding boxes around say the cars and the objects here because we need to track them and we need to understand how they might move around so again we might ask human annotators to give us some annotations for these and humans might go in and might tell you that ok those patterns over there are cars and bicycles and so on and you can train your neural network on this, but if you're not careful, the neural network all will make miss predictions in some cases."

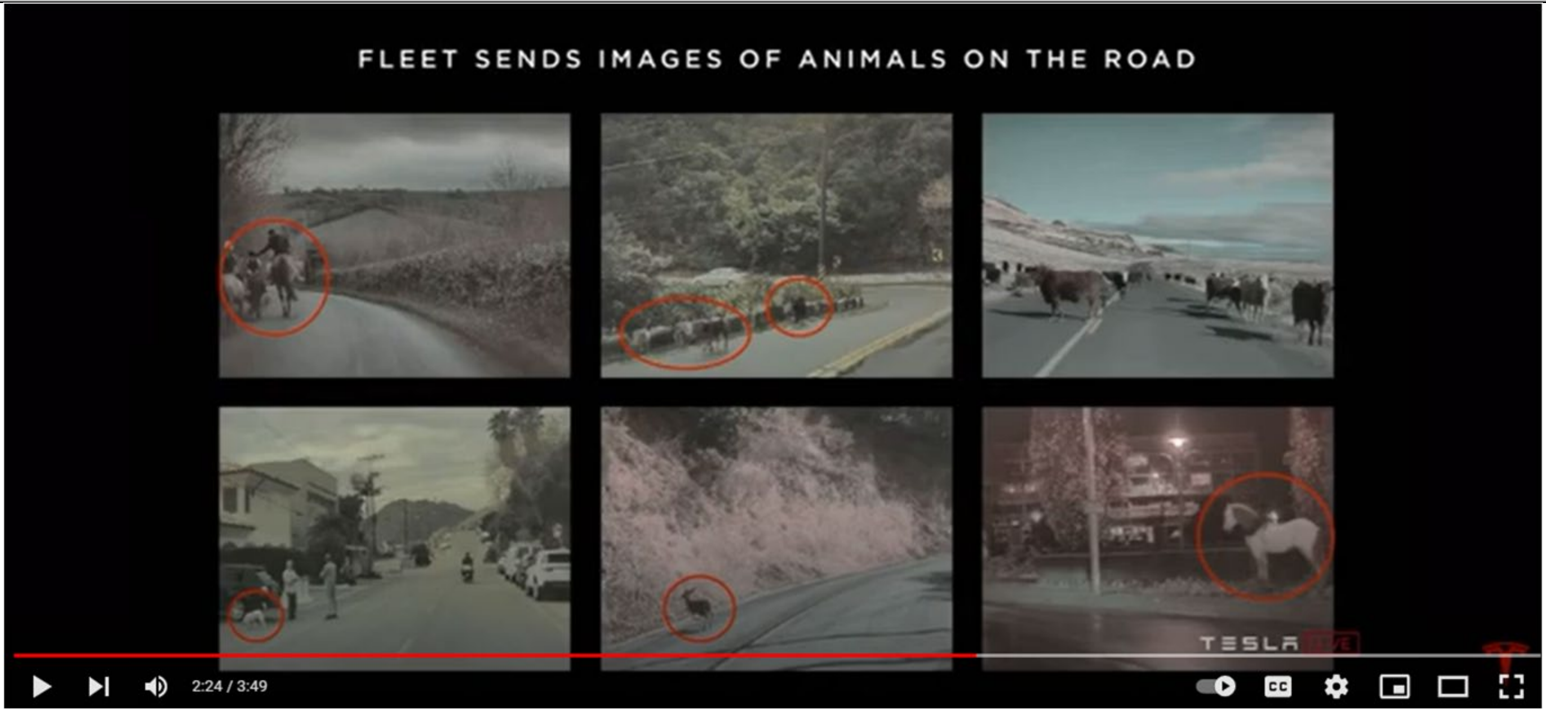
<table>
<tr><th>U.S. Patent No. 11,663,474</th><th>Exemplary Evidence of Plaintiff’s Theory(ies) of Infringement[1] against the’474 Accused Products[2]</th></tr>
<tr><td></td><td>


“so as an example if we stumble by a car like this that has a bike on the back of it then the neural network actually when I joined would actually create two deductions it would create a car deduction and a bicycle deduction and that’s actually kind of correct because I guess both of those objects actually exist but for the purposes of the controller and a planner downstream you really don’t want to deal with the fact that this bicycle can go with the car the truth is that that bike is attached to that car so in terms of like just objects on the road there’s a single object a single car and so what you’d like to do now is you’d like to just potentially annotate lots of those images as this is just a single car so the process that we that we go through internally in the team is that we take this image or a few images that show this pattern and we have a mechanism a machine learning mechanism by which we can ask the fleet to source us examples that look like that and the fleet might respond with images that contains those patterns [emphasis added]”
</td></tr>
</table>

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 "so as an example these six images might come from the fleet they all contain bikes on backs of cars and we would go in and we would annotate all those as just a single car and then the the performance of that detector actually improves and the network internally understands that hey when the bike is just attached to the car that's actually just a single car and it can learn that given enough examples and that's how we sort of fix that problem … now the fleet doesn't just respond with bicycles on backs of cars we look for all the thing we look for lots of things all the time."

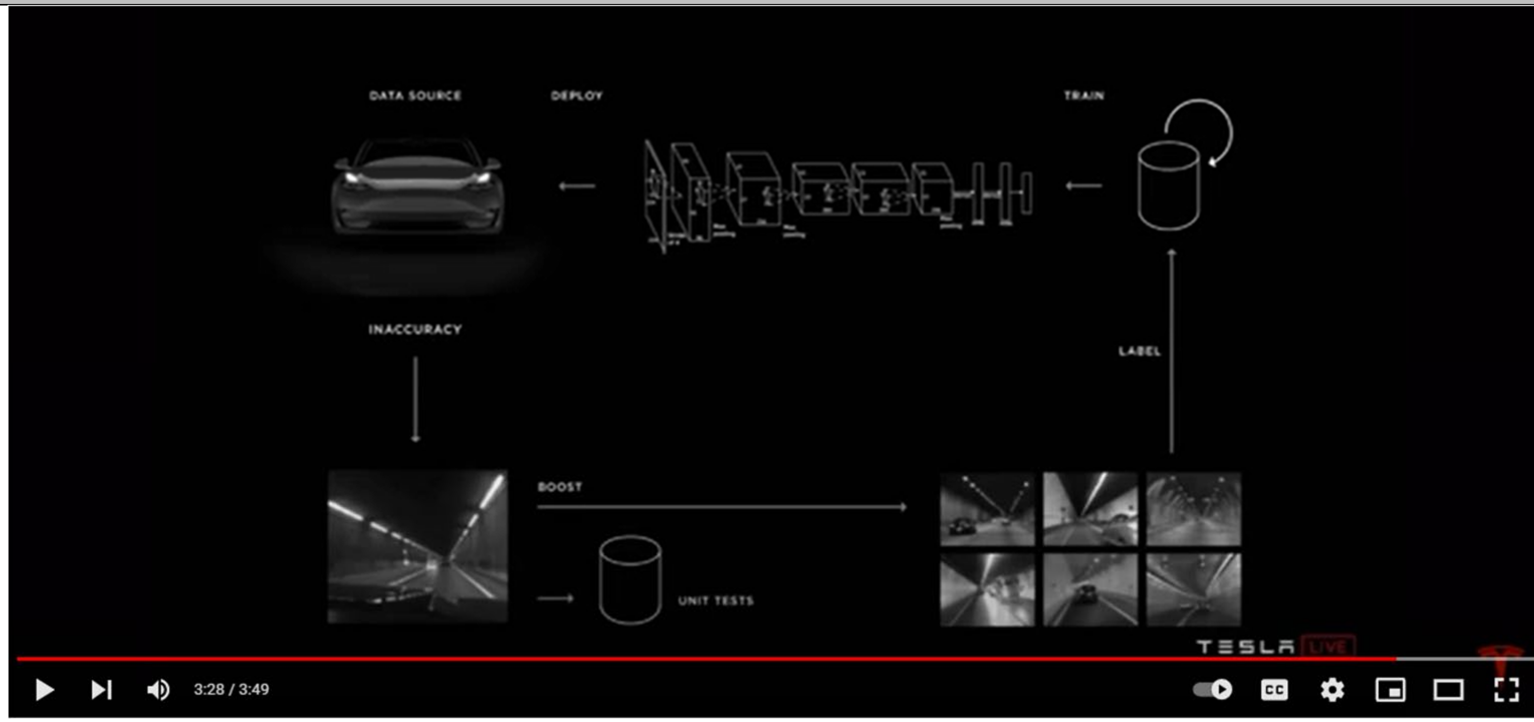
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 "so for example we look for boats and the fleet can respond with boats we look for construction sites and the fleet can send us lots of construction sites from across the world we look for even slightly more rare cases."

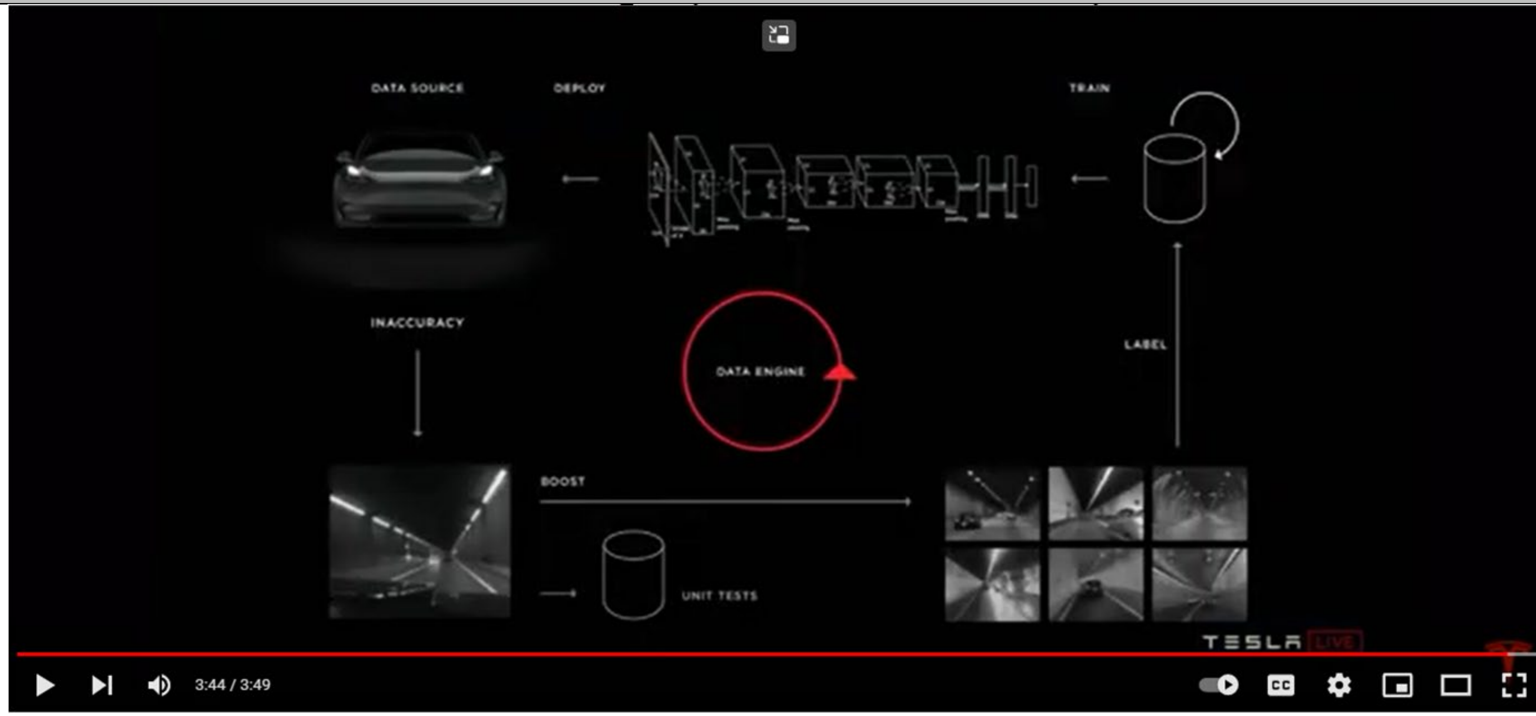
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 "so for example finding debris on the road is pretty important to us so these are examples of images that have streamed to us from the fleet that show tires cones, plastic bags and things like that if we can source these at scale we can annotate them correctly and *the neural network will learn how to deal with them in the world* [*emphasis* added]."

<table>
<tr><th>U.S. Patent No. 11,663,474</th><th>Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]</th></tr>
<tr><td></td><td>

"here's another example, animals of course also a very rare occurrence, an event, but we wanted them neural network to really understand what's going on here…that these are animals and we want to deal with that correctly."</td></tr>
</table>

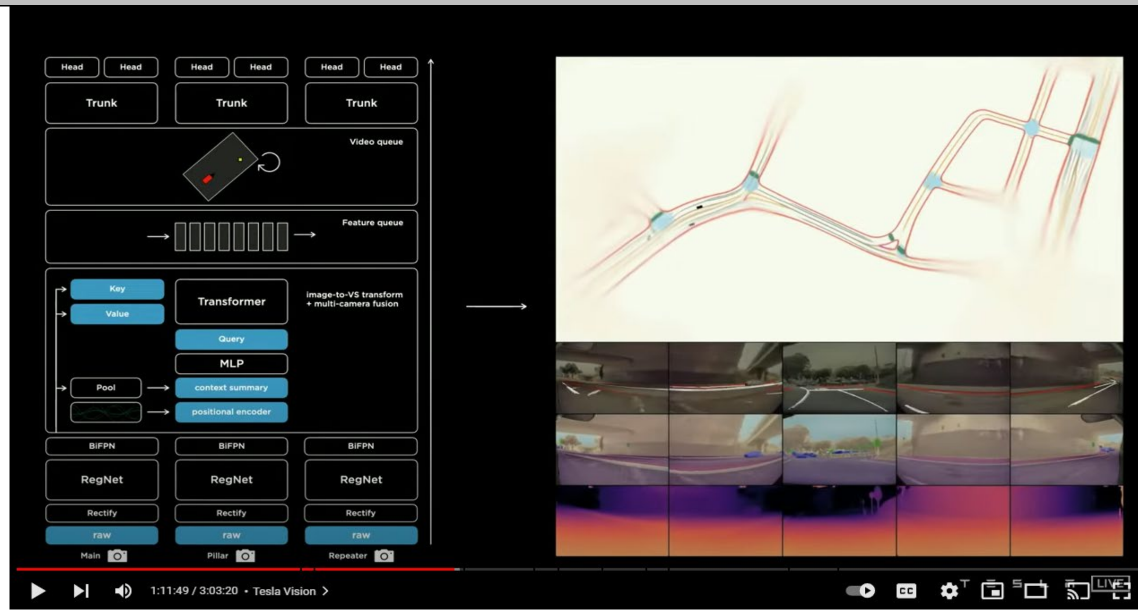
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 "so to summarize the process by which we iterate on neural network predictions looked something like this – we start with a seed dataset that was potentially sourced at random, we annotate that dataset, and then we train your networks on that dataset and put that in the car, and then we have mechanisms by which we notice inaccuracies in the car when this detector may be misbehaving."

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 "so, for example, if we detect that the neural network might be uncertain or if we detect that or *if there's a driver intervention or any of those settings we can create this trigger infrastructure that sends us data of those inaccuracies and so, for example, if we don't perform very well on lane line detection on tunnels then we can notice that there's a problem in tunnels that imagine would enter our unit tests so we can verify that we've actually fixing the problem over time but now what you do is to fix this inaccuracy you need to source many more examples that look like that so we asked the fleet to please send us many more tunnels and then we label all those tunnels correctly, we incorporate that into the training set and we retrain the network, redeploy, and iterate the cycle over and over again* [*emphasis* added]."

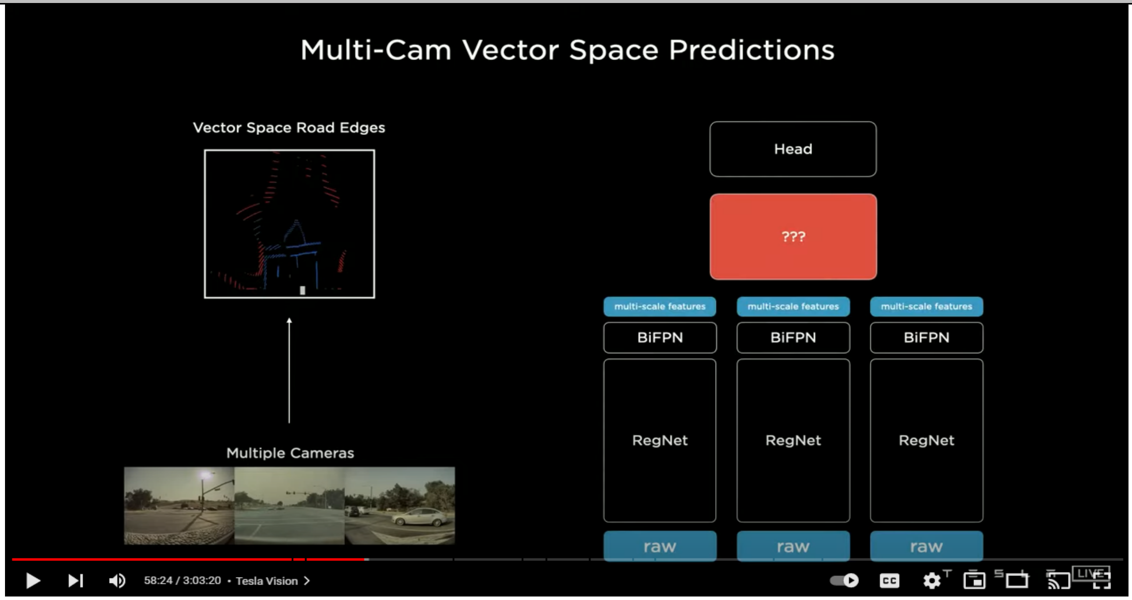
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 "so we refer to this iterative process by which we improve these predictions as the *data engine so iteratively deploying something potentially in shadow mode sourcing inaccuracies and incorporating the training set over and over again and we do this basically for all the predictions of these neural networks* [*emphasis* added]." Source: https://www.youtube.com/watch?v=33K3id2xNAE&t As Mr. Karpathy explained during Tesla Autonomy Day 2019 "while you are driving a car what you're actually doing is you are annotating the data because you are steering the wheel. you're telling us how to traverse different environments so what we're looking at here is a some person in the fleet who took a left through an intersection and what we do here is we have the full video of all the cameras and we know that the path that this person took because of the GPS, the inertial measurement unit, the wheel

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	angle, the wheel ticks, so we put all that together and we understand the path that this person took through this environment and then of course we can use this for supervision [e.g., training a CNN through supervised learning] for the network so we just source a lot of this from the fleet, we train a neural network on those trajectories, and then the neural network predicts paths just from that data so really what ***this is referred to typically is called imitation learning we're taking human trajectories from the real world I'm just trying to imitate how people drive in real worlds and we can also apply the same data engine crank to all of this and make this work*** [***emphasis*** added]." Source: https://www.youtube.com/watch?v=-b041NXGPZ8 at 1:04:10. In this way, driving knowledge is: (i) learned on a first Tesla vehicle, (ii) transferred to the simulator where it is synthesized with other driving knowledge, (iii) packaged and sent from the simulator to the fleet, e.g., the driving knowledge is stored on a memory of a second Tesla vehicle via the fleet over-the-air (OTA) software update, and (iv) accessed on the second Tesla vehicle for its autonomous driving. The second Tesla vehicle can, thereby, implement the driving knowledge learned by the first Tesla vehicle, synthesized by the supercomputer/simulator, and distributed to vehicles in Telsa's fleet over-the-air. *See also,* Tesla AI Day 2021 video (https://www.youtube.com/watch?v=j0z4FweCy4M at 2:55:29), during which Mr. Musk explained that all the human drivers are essentially training the neural network as to what is the correct course of action for the accused Tesla vehicles to take while being operated on the road. This is just one example of the system (e.g., the Tesla/Tesla fleet plus simulator) being trained with object representations and instructions from a vehicle. *See also*, Claim 3 of US 11, 238, 344, which states that, "…the first circumstance representation includes a first one or more object representations." Claim 18 of the '344 patent further explains that a circumstance representation includes, "…one or more data about a circumstance of a vehicle/device…" Additionally, the description of Fig. 4A (vehicle's/device's circumstance(s) including multiple detected objects), Fig. 4B (object representations including object properties such as distance, bearing/direction/coordinates, identity, type, computer model [e.g., bounding box, point cloud, etc.], etc.), and Fig. 38 (vehicle's/device's circumstance including multiple detected objects) add more generalized framework for the relationship between circumstance representations and object representations. The '344 patent provides examples of object properties that are of importance to the '474 Accused Products. These object properties include, "existence of an object, type of an object (i.e. person, cat, vehicle, building, street, tree, rock, etc.), identity of an object (i.e. name, identifier, etc.), distance of an object, bearing/angle of an object, location of an object

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	(i.e. distance and bearing/angle from a known point, coordinates, etc.), shape/size of an object (i.e. height, width, depth, computer model, point cloud, etc.), activity of an object (i.e. motion, gestures, etc.), and/or other properties of an object." (*See* the '344 patent at 72:44-52). *See also*, Tesla AI Day 2021(https://www.youtube.com/watch?v=j0z4FweCy4M) for additional examples of Tesla's analysis of objects and circumstances.  At 58:10, the speaker describes taking all of the images and simultaneously feeding them into a single neural net and directly outputting into a vector space.

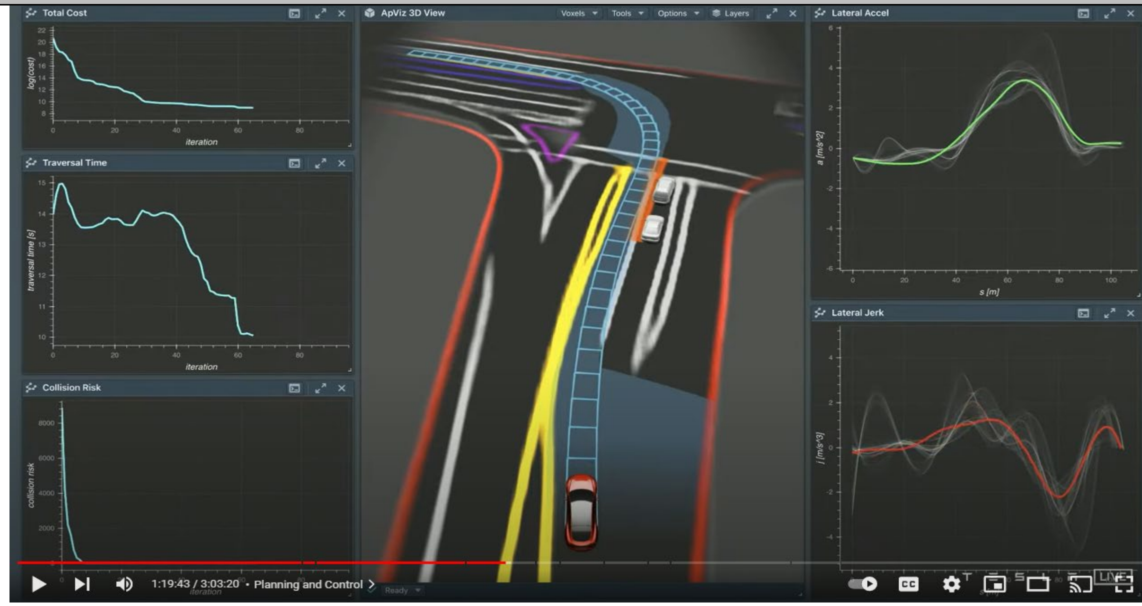
<table>
<tr><th>U.S. Patent No. 11,663,474</th><th>Exemplary Evidence of Plaintiff’s Theory(ies) of Infringement[1] against the’474 Accused Products[2]</th></tr>
<tr><td></td><td>


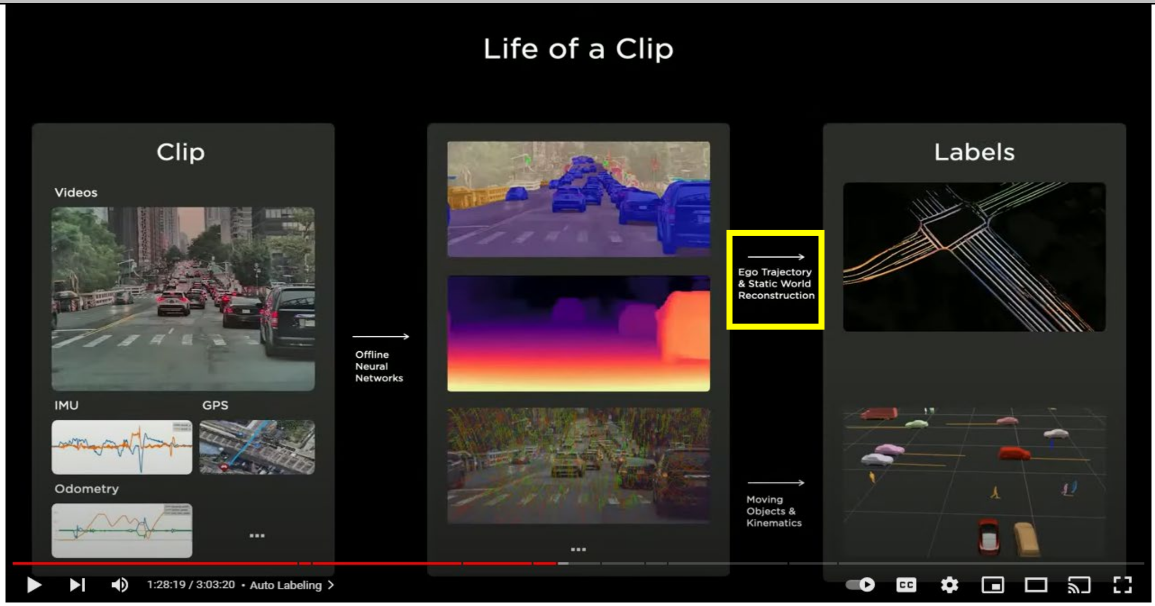
Around 1:11:20, the speaker begins describing how Tesla uses a feed of raw images captured by the cameras included in the accused Tesla vehicles, and how the images captured by the cameras pass through a transformer module to re-represent the objects captured in the images taken by the camera into the vector space.
</td></tr>
</table>

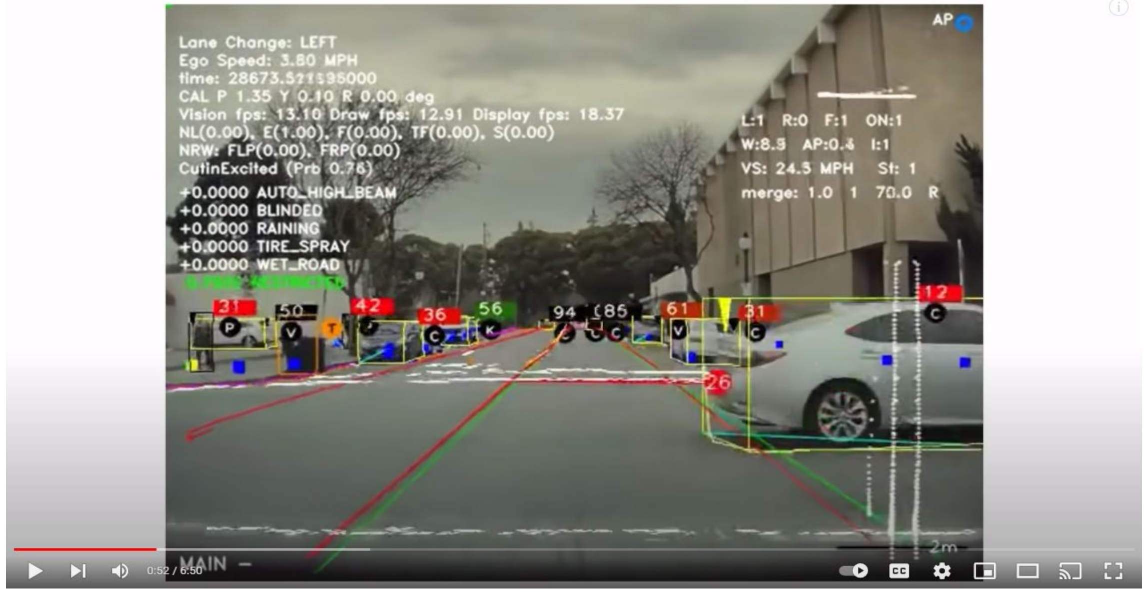
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 At 1:31:23 the speaker describes Tesla's ability to arbitrarily reconstruct 3D static obstacles, and the slide shown in the screenshot above depicts an example of Tesla's reconstructed 3D point cloud. Tesla is able to make this reconstruction from data captured by the cameras and other various sensors included in the accused Tesla vehicles.

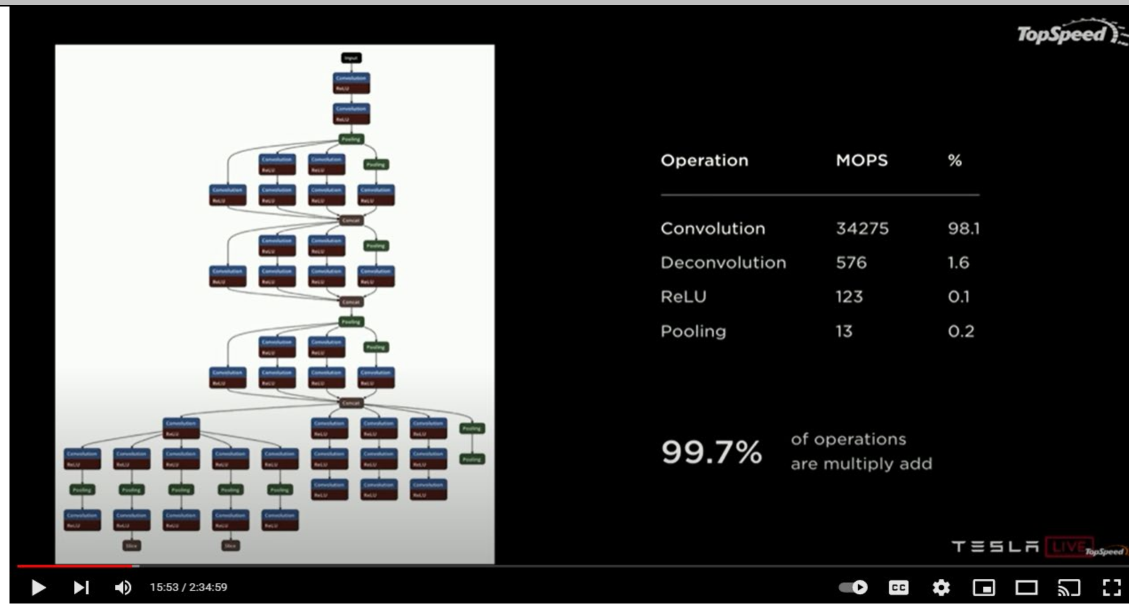
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 At 1:31:58 the speaker is describing Tesla's vector space in more detail, notably touching on how the vector space includes object representations.

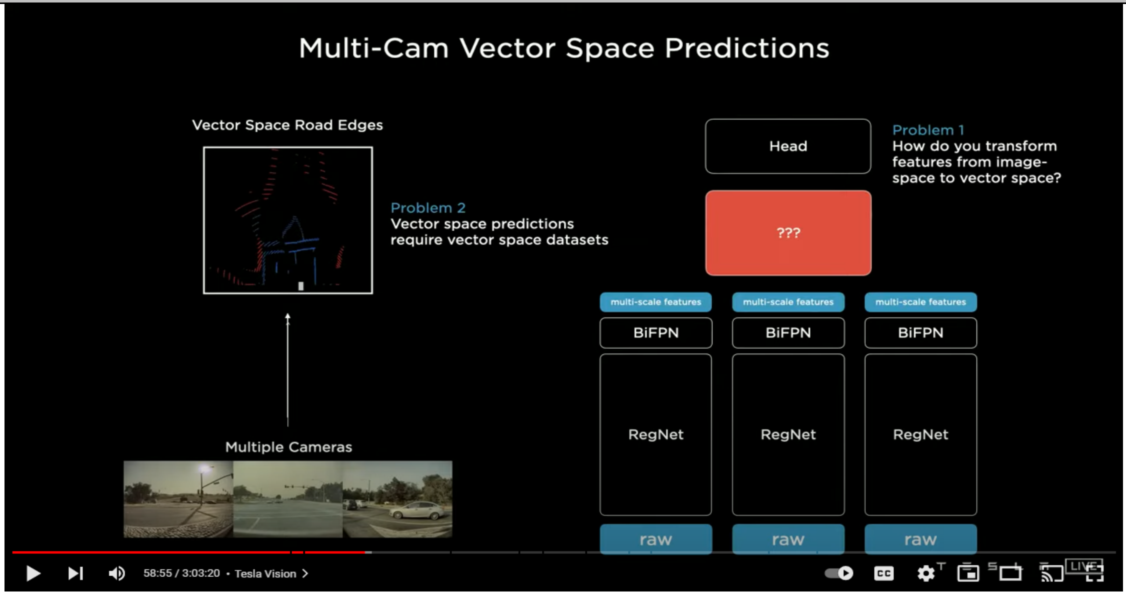
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 At 1:32:48, the speaker explains how Tesla combines everything together in order to produce "amazing" data sets that annotate all of the road texture, the static objects, and all of the moving objects. For additional examples of Tesla's correlating a first circumstance representation with one or more instruction sets, *see also*, Tesla AI Day 2021(https://www.youtube.com/watch?v=j0z4FweCy4M)

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 At 1:19:43, the speaker describes recording a human's trajectory around a curve, recording/storing the data associated therewith, and sharing that data with the fleet so the accused Tesla vehicles are better able to handle that particular turn and similar turns/curves in the future.

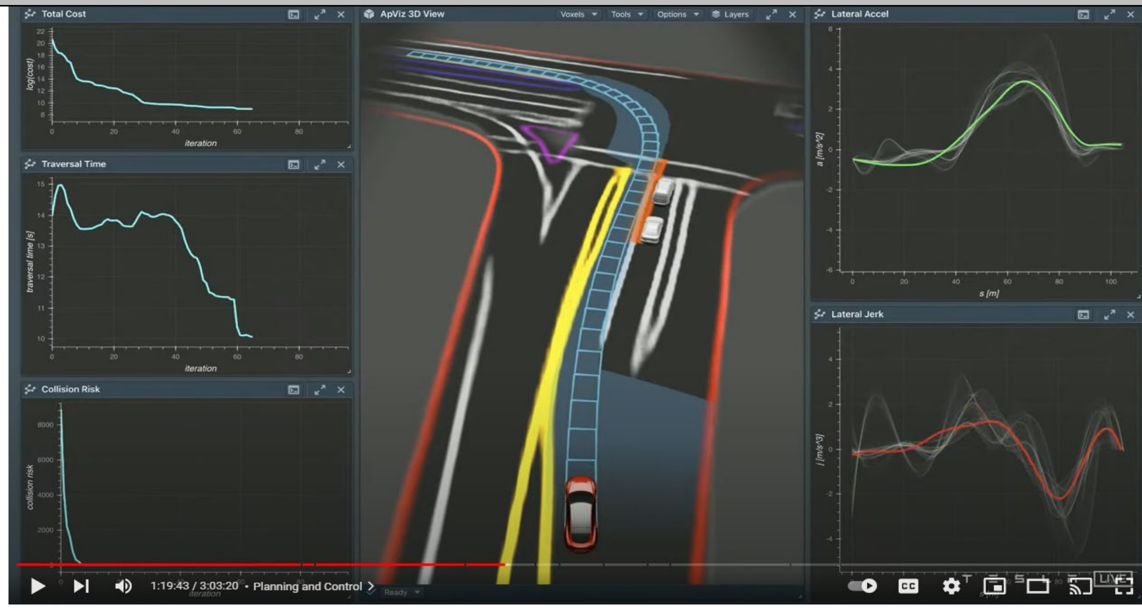
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 At/around 1:28:16, the speaker is describing exemplary labeling of a single clip in which an entity that has dense sensor data like videos, imu data, gps, odometry (e.g., speed, steering braking, etc.) are uploaded by Tesla's own engineering cars and/or from customer cars. Note the "Ego Trajectory & Static World Reconstruction" label shown in the screenshot above (yellow annotation). Again, at/around 2:55:29, Mr. Musk describes how all the human drivers are "essential training the neural net as to what is the correct course of action" in any particular circumstance encountered by the accused Tesla vehicles. A further example of Tesla's correlating a first circumstance representation with one or more instruction set is depicted below.

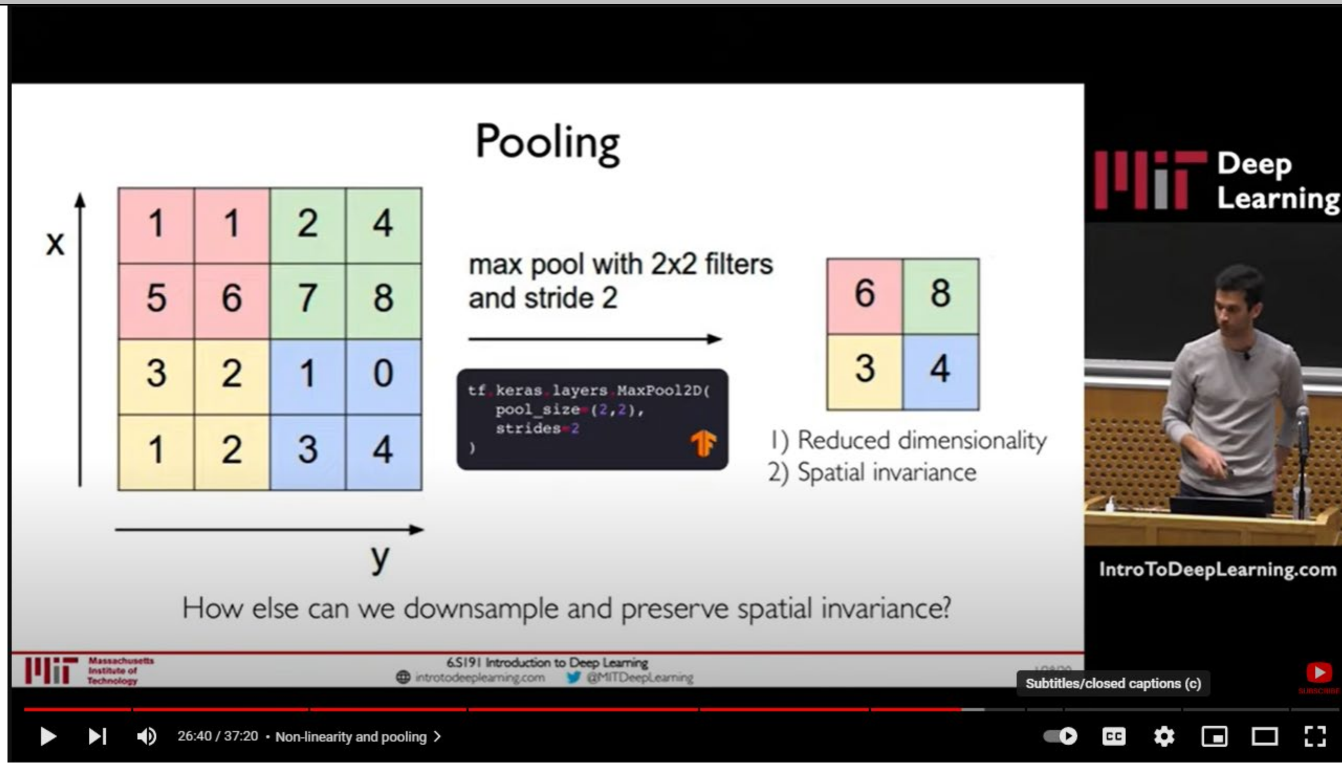
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 At/around 0:50 in the video referenced above/cited below, a myriad of instruction labels considered by the accused Tesla vehicles while in operation can be seen. Examples of such instruction labels include Lane change and Ego speed. Source: https://www.youtube.com/watch?v=zRnSmw1i_DQ Tesla describes how circumstance representations are correlated with various instruction sets for operation of the accused Tesla vehicles during Tesla Autonomy Day 2019 (https://www.youtube.com/watch?v=-b041NXGPZ8) and Tesla AI Day 2021 (https://www.youtube.com/watch?v=j0z4FweCy4M). As examples:

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 Tesla describes how the accused Tesla vehicles use a convolutional neural network (CNN) to learn and implement autonomous driving. (*See* Tesla Autonomy Day 2019 at/around 15:53).

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 Tesla describes how it wants to re-represent the features from image space as vector space features, and that it wants to use vector space predictions from its neural network, which requires vector space data sets, and vector space data sets require vector space labels. (*See* Tesla AI Day 2021 at/around 58:31).

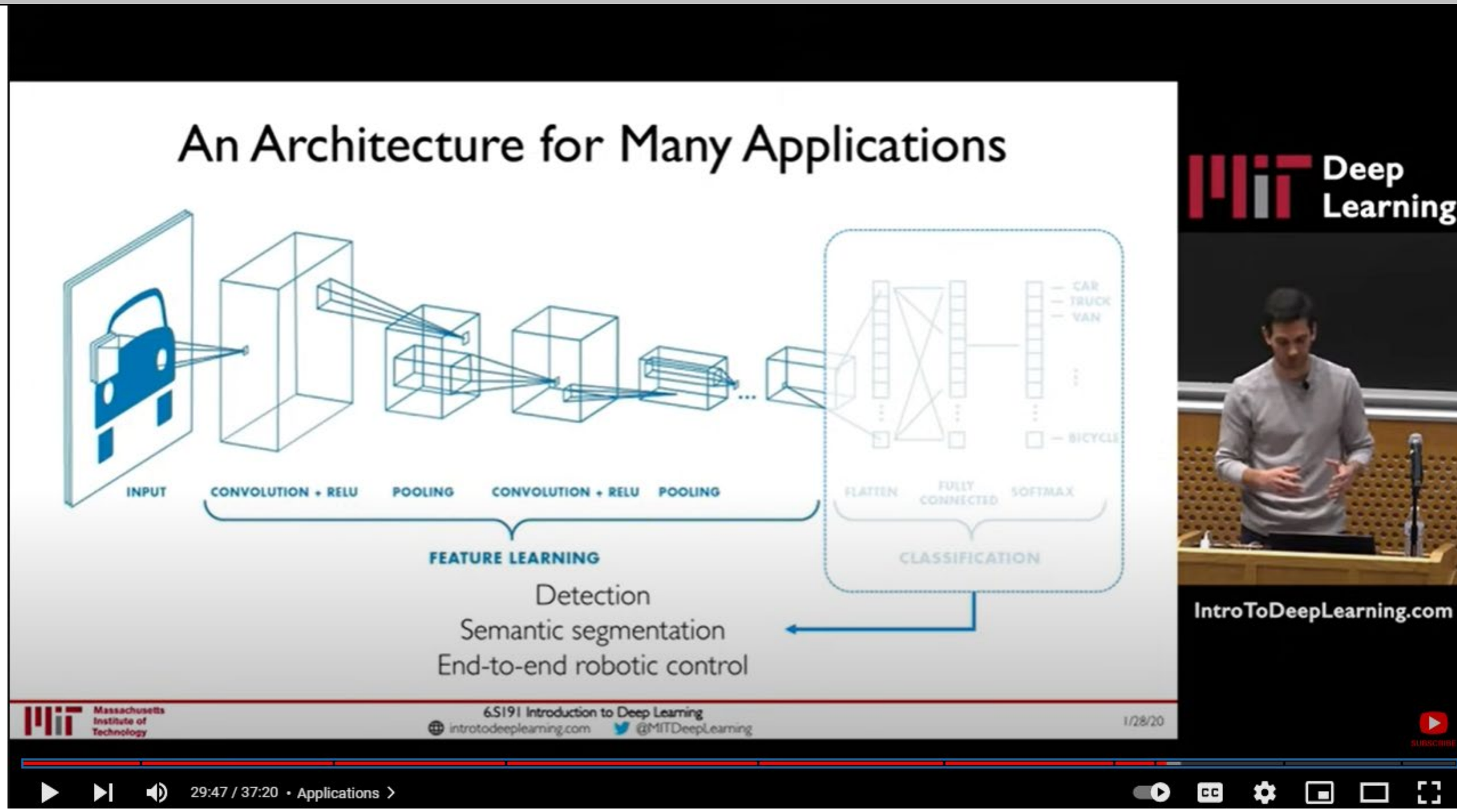
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 Tesla describes how, while driving accused Tesla vehicles, drivers are actually annotating the data captured by the vehicle because the user is steering the wheel, thereby telling the vehicle/Tesla how to navigate/traverse different environment (e.g., circumstance representations representing the environment correlated with driving instructions for traversing the environment). (*See* Tesla Autonomy Day 2019 at/around 1:04:10)

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 Tesla depicts circumstance representation and correlated human trajectory, e.g., trajectory implemented using driving instruction sets at each point in time while the accused Tesla vehicles are taking a turn. (*See* Tesla AI Day 2021 at/around 1:19:43). Further, as stated above, circumstance representations include one or more object representations/features. The convolution layers of the CNN in the accused Tesla vehicles store object features correlated via forward connections with output nodes of the CNN, including instructions for driving the vehicle, as described by the expert from MIT speaking in the following video, the "MIT Convolutional Neural Networks" video, found at: https://www.youtube.com/watch?v=iaSUYvmCekI. First, the speaker describes learning features and features being stored/learned in different convolution layers.

<table>
<tr><th>U.S. Patent No. 11,663,474</th><th>Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]</th></tr>
<tr><td></td><td>

(See MIT Convolutional Neural Networks video at/around 26:40).

Next, the speaker describes the overall CNN used for robotic control (e.g., autonomous driving) where features in convolutional layers of the CNN are correlated with control instructions at the end of the CNN via forward connections.</td></tr>
</table>

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 (*See* MIT Convolutional Neural Network video at/around 29:40) During its Autonomy Day 2019 and AI Day 2021 presentations, Tesla goes on to describe how the instruction sets for operating the device is learned in a learning process that includes operating the first device at least partially by a user. For example:

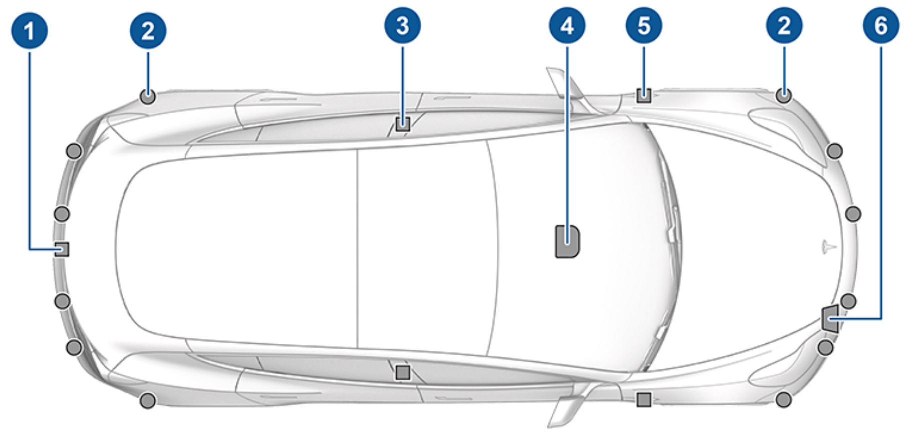
<table>
<tr><th>U.S. Patent No. 11,663,474</th><th>Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]</th></tr>
<tr><td></td><td>

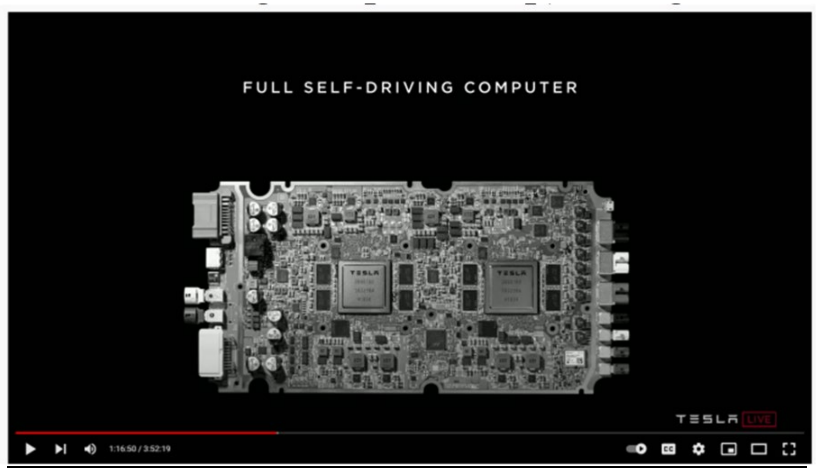
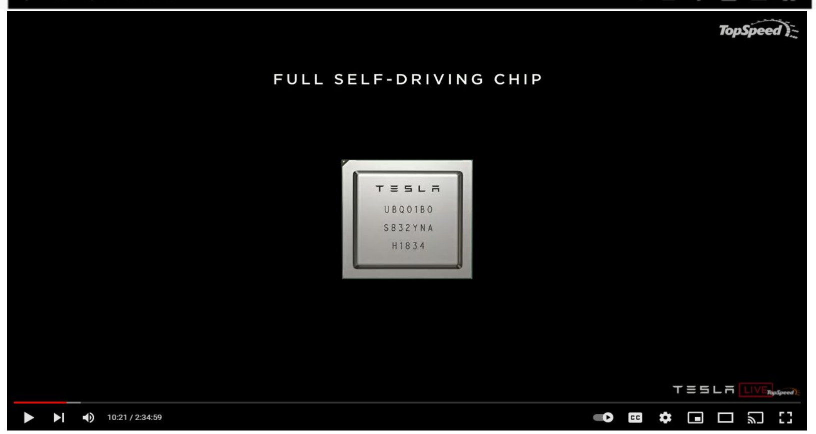
At/around 1:04:10 in Tesla's Autonomy Day 2019 presentation, the speaker describes, while driving one of the accused Tesla vehicles, the user is annotating the data because the user is steering the wheel, the user is telling Tesla how to traverse different environments, that one person in the fleet taking, e.g., a left turn through a particular intersection, is providing Tesla with invaluable data corresponding to GPS coordinates, inertial measurements, wheel angle, wheel ticks, etc. thus enabling Tesla to understand the path that particular user took through that particular turn. Tesla then uses this data to supervising/training the CNN (e.g., training through supervised learning). This data is then shared with the entire fleet through the neural network. The speaker goes on to state that Tesla is merely trying to imitate how people drive in the real world.

At/around 1:04:01 in Tesla's Autonomy Day 2019 presentation, the speaker describes how Teslas/Tesla's neural network learns from drivers training the network. At/around1:52:20, it is stated human drivers are monitored (e.g., monitored by a camera), and that the CNN implements autonomous driving based on this learned knowledge.

See also Tesla's US 10,956,755; Tesla's US 2020/0249685; Tesla's US 10,997,461; Tesla's WO 2020/056331.</td></tr>
</table>

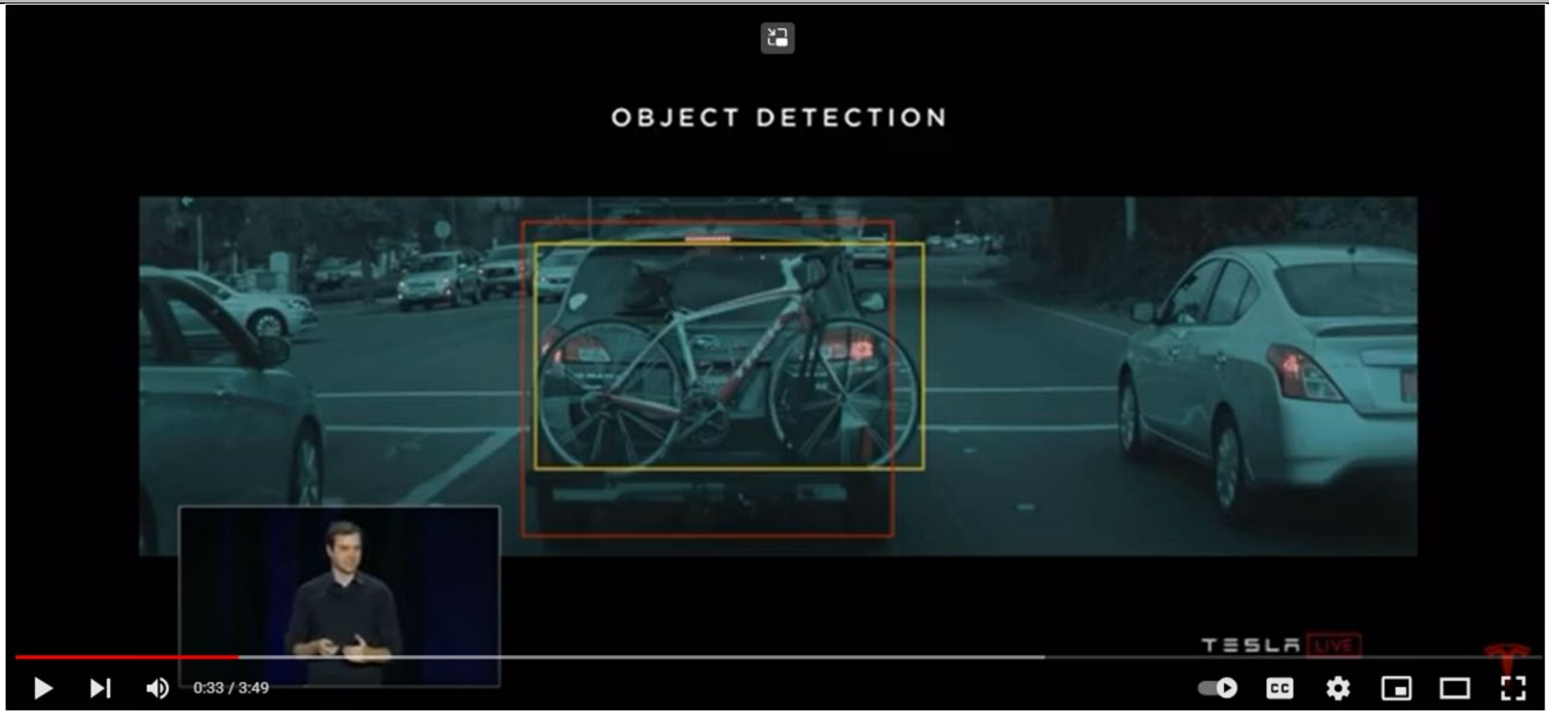
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
[45b]. one or more sensors;	The '474 Accused Products are devices further comprising one or more sensors. For example, the 474 Accused Products further includes numerous sensors. For instance, each vehicle in the Tesla fleet of vehicles includes at least an array of eight (8) outward facing cameras that capture the environment surrounding the vehicle. Optionally, ultrasonic sensors and radar can be equipped. Data detected/sensed by these sensors is provided to accused Tesla autonomous vehicle simulation system, e.g., via internet uplink.

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	TESLA | Model 3 Owner's Manual Model 3 Owner's Manual Overview Opening and Closing Storage Areas Seating and Safety Restraints Connectivity Driving Autopilot **About Autopilot** Traffic-Aware Cruise Control Autosteer Navigate on Autopilot Traffic Light and Stop Sign Control Autopark Summon Smart Summon Active Safety Features Dashcam, Sentry, and Security Climate Navigation and Entertainment Charging and Energy Consumption Maintenance Specifications Roadside Assistance Troubleshooting Consumer Information ## How It Works Your Model 3 includes the following components that actively monitor the surrounding area:  1. A camera is mounted above the rear license plate. 2. Ultrasonic sensors (if equipped) are located in the front and rear bumpers. 3. A camera is mounted in each door pillar. 4. Three cameras are mounted to the windshield above the rear view mirror. 5. A camera is mounted to each front fender. 6. Radar (if equipped) is mounted behind the front bumper. Model 3 is also equipped with high precision electronically-assisted braking and steering systems. In addition, the cabin camera (if equipped) can determine driver inattentiveness and provide alerts when Autopilot is engaged. By default, data from the camera does not leave the vehicle itself. In other words, data is not saved or transmitted unless you enable data sharing. To enable data sharing, touch **Controls > Software > Data Sharing > Allow Cabin Camera Analytics**. Cabin Camera Analytics helps Tesla continue to develop even safer vehicles in the future. See Cabin Camera. Source: https://www.tesla.com/ownersmanual/model3/en_au/GUID-EDA77281-42DC-4618-98A9-CC62378E0EC2.html#:~:text=Ultrasonic%20sensors%20(if%20equipped)%20are,mounted%20to%20each%20front%20fender.

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
[45c]. one or more processors; and	The '474 Accused Products are devices further comprising one or more processors. For example, Tesla's Models S, 3, X, and Y include one or more processors that, at a minimum, effect semi and/or fully autonomous driving.   Source: https://www.youtube.com/watch?v=-b041NXGPZ8 (Tesla Autonomy Day 2019, discussing Tesla's full self-driving computer and Tesla's full self-driving chip); *see also* Tesla's ECU, which contains computers that are programmed to perform the claimed features.

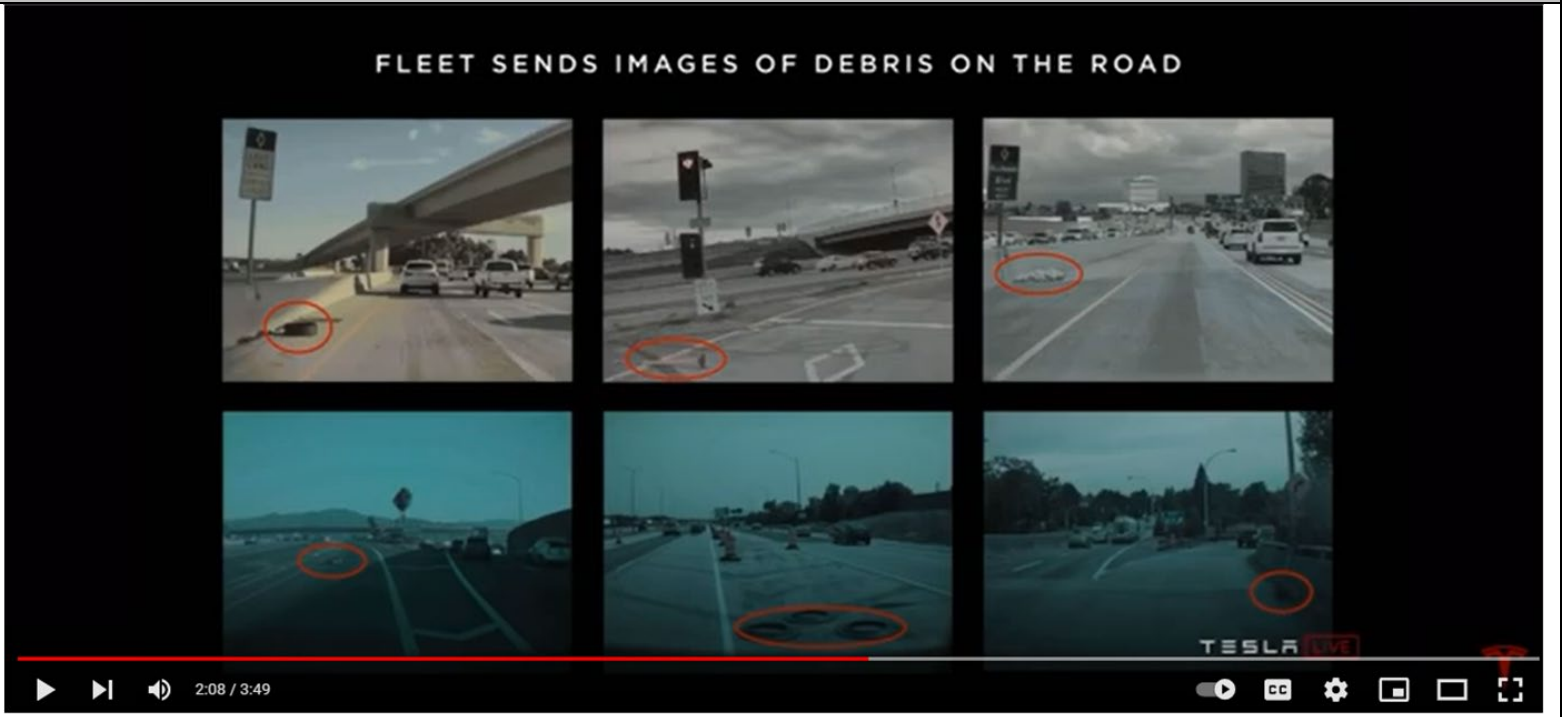
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
[45d]. one or more non-transitory machine readable media storing machine readable code that, when executed by the one or more processors, causes the one or more processors to perform at least:	The '474 Accused Products are devices further comprising one or more non-transitory machine readable media storing machine readable code that, when executed by the one or more processors, causes the one or more processors to perform at least the following limitations. For example, each accused Tesla vehicle includes a system(s) that access memory that stores representations of objects in the surroundings of the Tesla vehicle. These circumstance representations (or circumstances) are learned while drivers drive the accused Tesla vehicles. Examples of circumstances that can be learned include, effectively everything captured/sensed/measured by the sensors (e.g., cameras, ultrasonic, telemetry, etc.) included in each of the accused Tesla vehicles that is fed into the ECU, such as other vehicles, pedestrians, roads, buildings, bikes, road debris, curbs, sidewalks, animals, speed, location, wheel ticks, etc. The '474 Accused Products also learn driving instructions, such as instructions for effecting the speed, steering, and/or braking of the accused Tesla vehicle being driven based on the actions taken by the driver to navigate those circumstances. This driving knowledge learned by one accused Tesla vehicle is transmitted to Tesla's simulator before being distributed to all vehicles in the Tesla fleet via over-the-air (OTA) software updates. Therefore, the fleet enables each accused Tesla vehicle included therein to autonomously implement driving instructions learned on one or more originating vehicles when similar circumstances are detected by any of the vehicles in Tesla's fleet, thereby enabling Tesla's autonomous driving capabilities. Further, to effectuate the circumstance representations, a Tesla vehicle detects objects. For example, in the previous analysis of driving behind the car with the bike mounted to it, the car detects the bike as an object. Tesla explains object detection as follows:

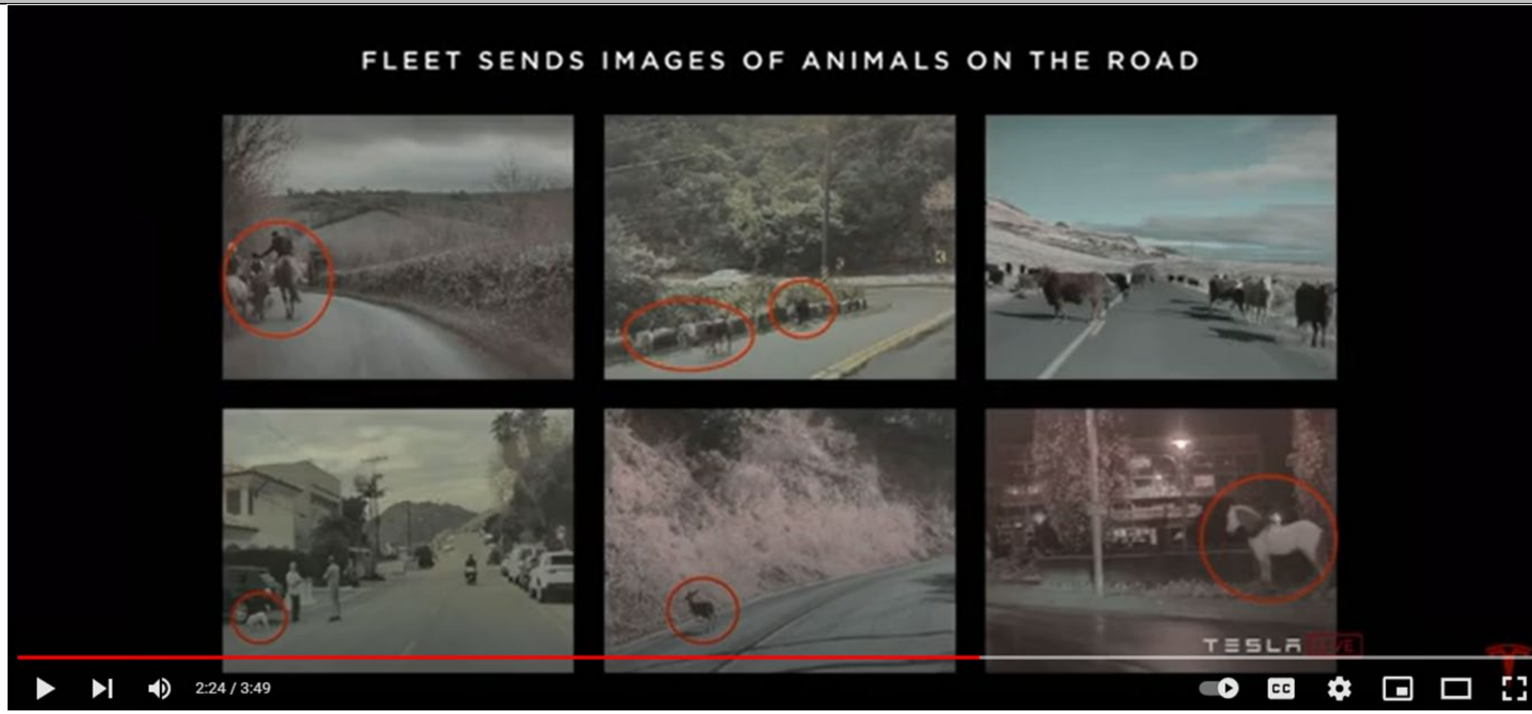
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 "So object detection is something we care a lot about we'd like to put bounding boxes around say the cars and the objects here because we need to track them and we need to understand how they might move around so again we might ask human annotators to give us some annotations for these and humans might go in and might tell you that ok those patterns over there are cars and bicycles and so on and you can train your neural network on this, but if you're not careful, the neural network all will make miss predictions in some cases."

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 "So as an example if we stumble by a car like this that has a bike on the back of it then the neural network actually when I joined would actually create two deductions it would create a car deduction and a bicycle deduction and that's actually kind of correct because I guess both of those objects actually exist but for the purposes of the controller and a planner downstream you really don't want to deal with the fact that this bicycle can go with the car the truth is that that bike is attached to that car so in terms of like just objects on the road there's a single object a single car and so what you'd like to do now is you'd like to just potentially annotate lots of those images as this is just a single car so the process that we that we go through internally in the team is that we take this image or a few images that show this pattern and we have a mechanism a machine learning mechanism by which we can ask the fleet to source us examples that look like that and the fleet might respond with images that contains those patterns."

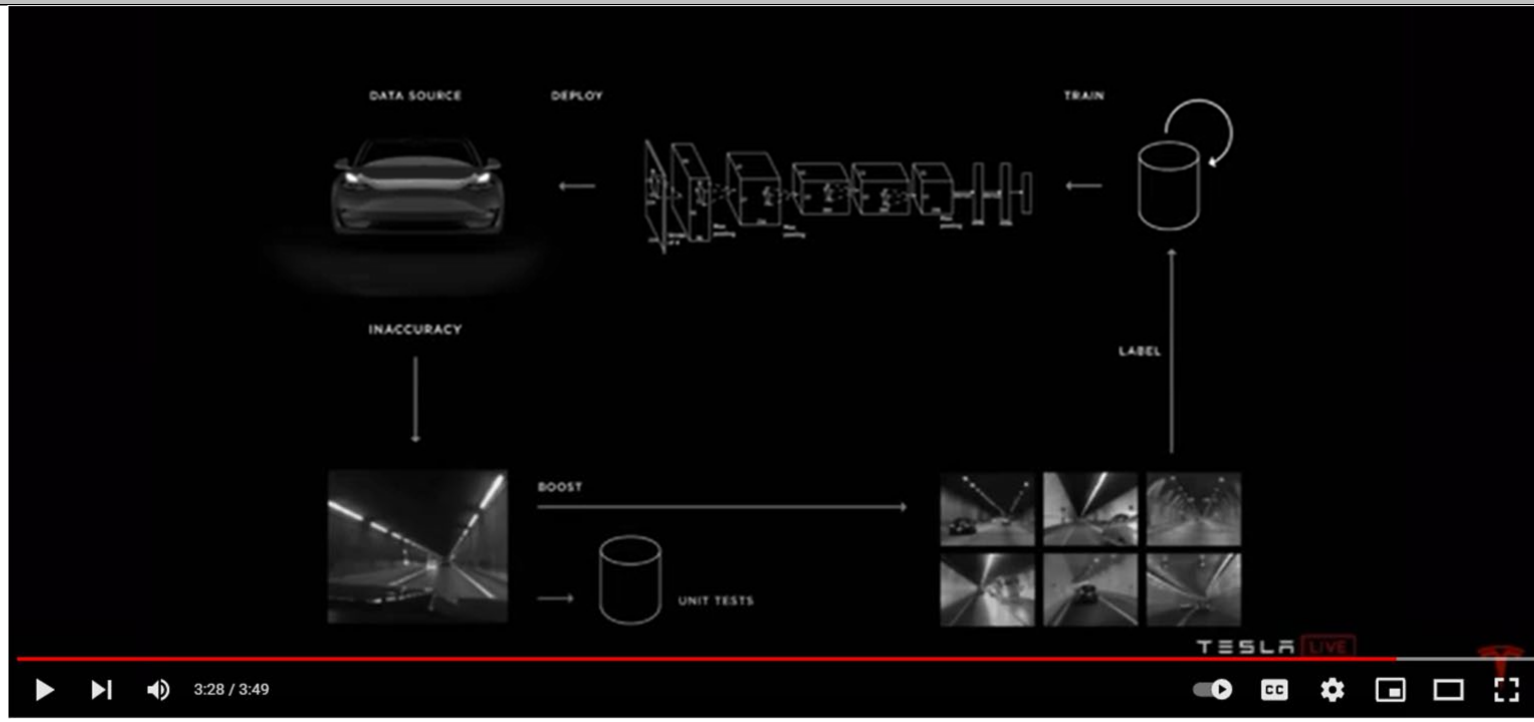
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 "So as an example these six images might come from the fleet they all contain bikes on backs of cars and we would go in and we would annotate all those as just a single car and then the performance of that detector actually improves and the network internally understands that hey when the bike is just attached to the car that's actually just a single car and it can learn that given enough examples and that's how we sort of fix that problem … now the fleet doesn't just respond with bicycles on backs of cars we look for all the thing we look for lots of things all the time."

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	  "So for example we look for boats and the fleet can respond with boats we look for construction sites and the fleet can send us lots of construction sites from across the world we look for even slightly more rare cases."

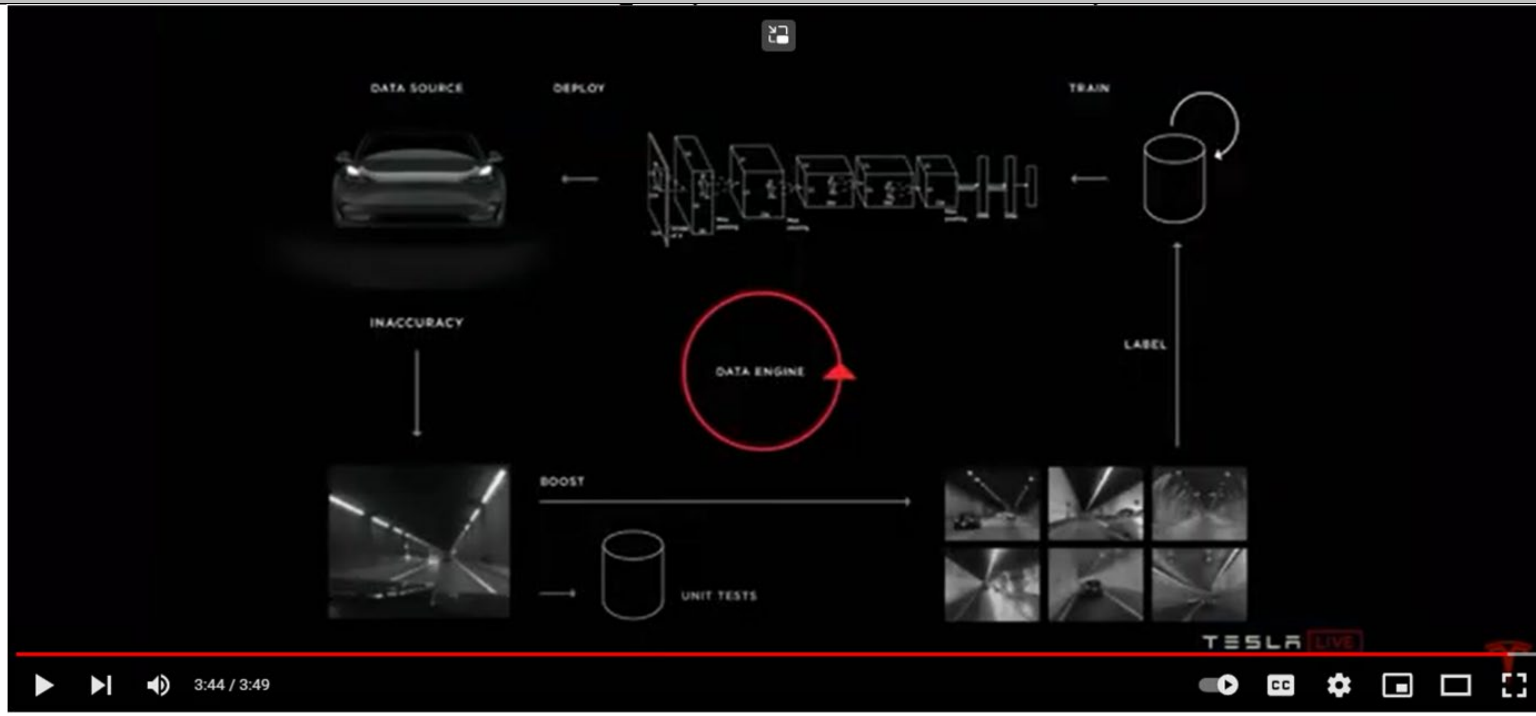
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 "So for example finding debris on the road is pretty important to us so these are examples of images that have streamed to us from the fleet that show tires cones, plastic bags and things like that if we can source these at scale we can annotate them correctly and the neural network will learn how to deal with them in the world."

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 "Here's another example, animals of course also a very rare occurrence, an event, but we wanted them neural network to really understand what's going on here…that these are animals and we want to deal with that correctly."

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 "So to summarize the process by which we iterate on neural network predictions looked something like this – we start with a seed dataset that was potentially sourced at random, we annotate that dataset, and then we train your networks on that dataset and put that in the car, and then we have mechanisms by which we notice inaccuracies in the car when this detector may be misbehaving."

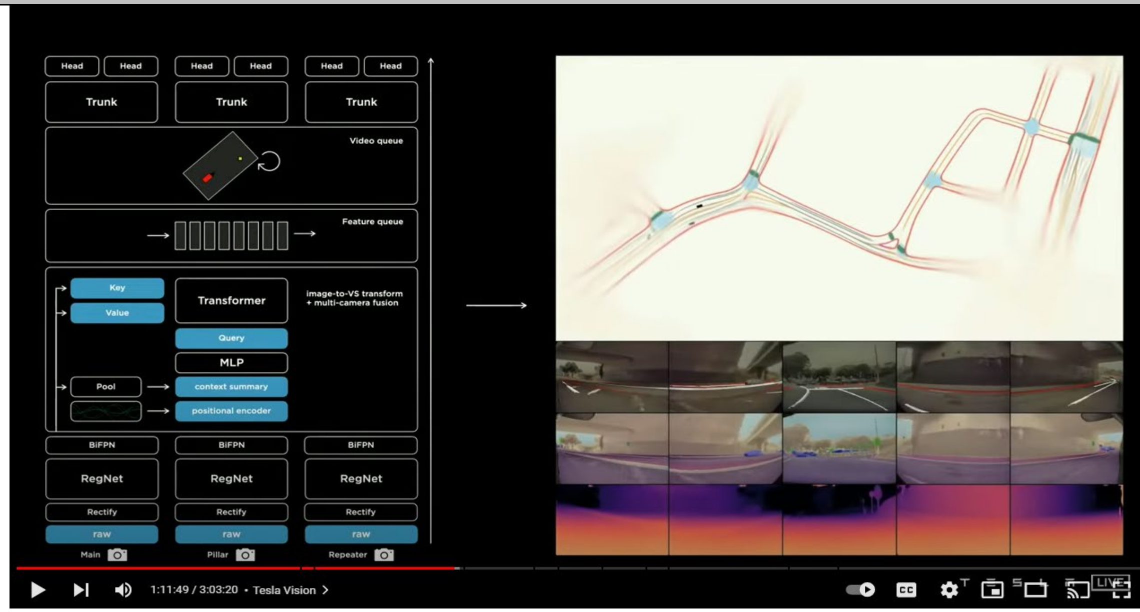
<table>
<tr><th>U.S. Patent No. 11,663,474</th><th>Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]</th></tr>
<tr><td></td><td>

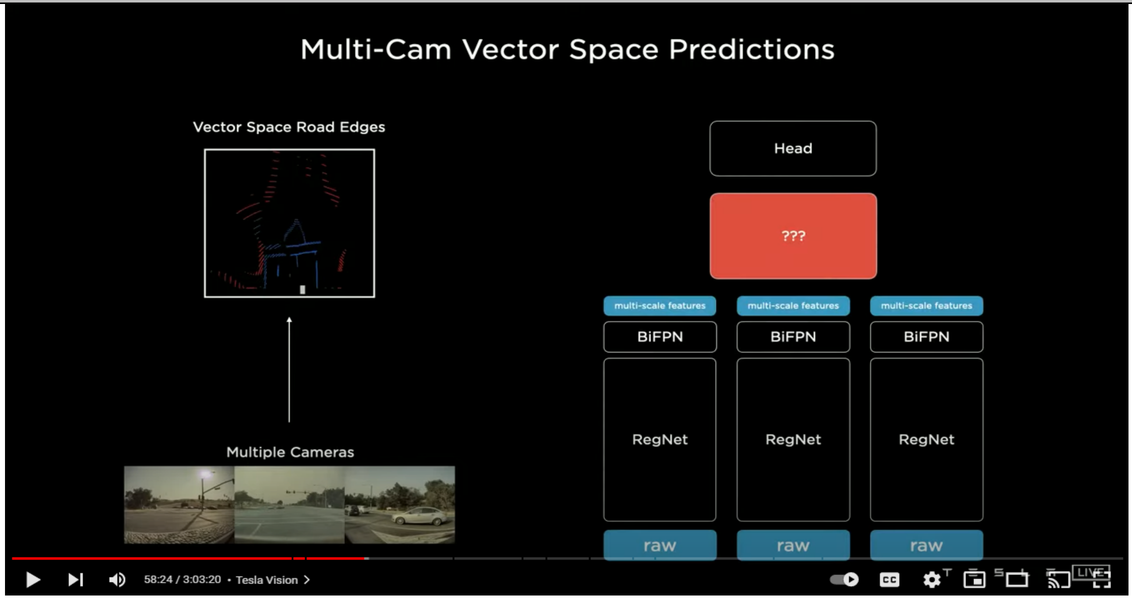
"So, for example, if we detect that the neural network might be uncertain or if we detect that or if there's a driver intervention or any of those settings we can create this trigger infrastructure that sends us data of those inaccuracies and so, for example, if we don't perform very well on lane line detection on tunnels then we can notice that there's a problem in tunnels that imagine would enter our unit tests so we can verify that we've actually fixing the problem over time but now what you do is to fix this inaccuracy you need to source many more examples that look like that so we asked the fleet to please send us many more tunnels and then we label all those tunnels correctly, we incorporate that into the training set and we retrain the network, redeploy, and iterate the cycle over and over again"</td></tr>
</table>

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 "So we refer to this iterative process by which we improve these predictions as the data engine so iteratively deploying something potentially in shadow mode sourcing inaccuracies and incorporating the training set over and over again and we do this basically for all the predictions of these neural networks." Source: https://www.youtube.com/watch?v=33K3id2xNAE&t As Mr. Karpathy explained during Tesla Autonomy Day 2019 "while you are driving a car what you're actually doing is you are annotating the data because you are steering the wheel. you're telling us how to traverse different environments so what we're looking at here is a some person in the fleet who took a left through an intersection and what we do here is we have the full video of all the cameras and we know that the path that this person took because of the GPS, the inertial measurement unit, the wheel angle, the wheel ticks, so we put all that together and we understand the path that this person took through this environment and then of course we can use this for supervision [e.g., training a CNN through supervised

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	learning] for the network so we just source a lot of this from the fleet, we train a neural network on those trajectories, and then the neural network predicts paths just from that data so really what this is referred to typically is called imitation learning we're taking human trajectories from the real world I'm just trying to imitate how people drive in real worlds and we can also apply the same data engine crank to all of this and make this work." Source: https://www.youtube.com/watch?v=-b041NXGPZ8 at 1:04:10. In this way, driving knowledge is: (i) learned on a first Tesla vehicle, (ii) transferred to the simulator where it is ynthesized with other driving knowledge, (iii) packaged and sent from the simulator to the fleet, e.g., the driving knowledge is stored on a memory of a second Tesla vehicle via the fleet over-the-air (OTA) software update, and (iv) accessed on the second Tesla vehicle for its autonomous driving. The second Tesla vehicle can, thereby, implement the driving knowledge learned by the first Tesla vehicle and synthesized by the simulator. *See also,* Tesla AI Day 2021 (https://www.youtube.com/watch?v=j0z4FweCy4M at or around 2:55:29), during which Mr. Musk explained that all the human drivers are essentially training the neural network as to what is the correct course of action for the accused Tesla vehicles to take while being operated on the road. This is just one example of the system (e.g., the Tesla/Tesla fleet plus the supercomputer system or other computers) being trained with object representations and instructions from a vehicle. *See also*, Tesla AI Day 2021(https://www.youtube.com/watch?v=j0z4FweCy4M) for additional examples of Tesla's analysis of objects and circumstances:

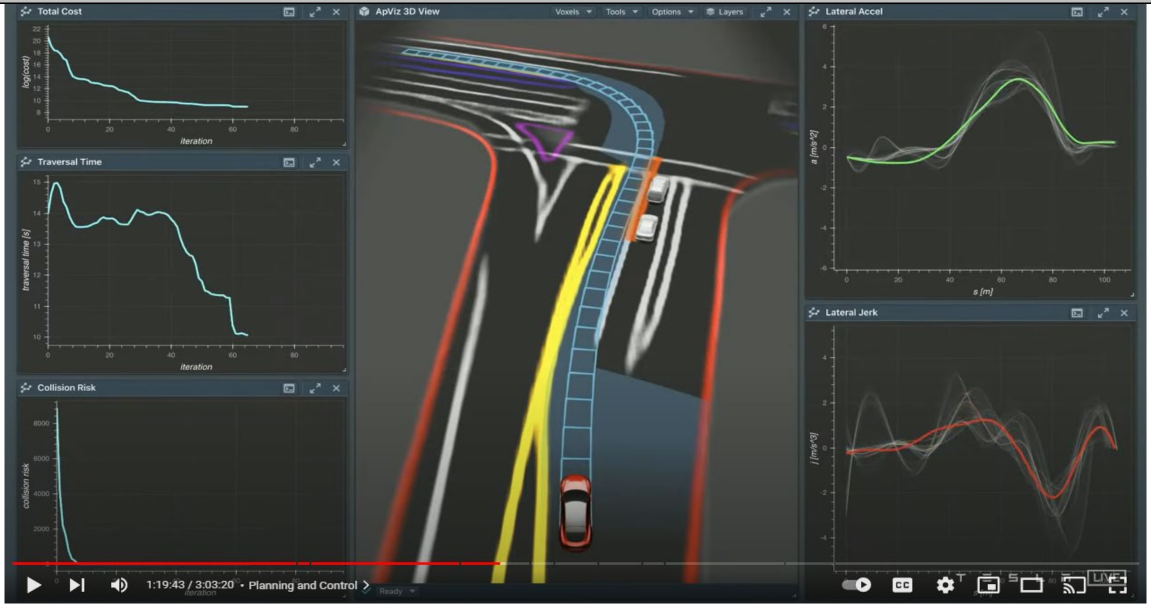
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 At 58:10, the speaker describes taking all of the images and simultaneously feeding them into a single neural net and directly outputting into a vector space.

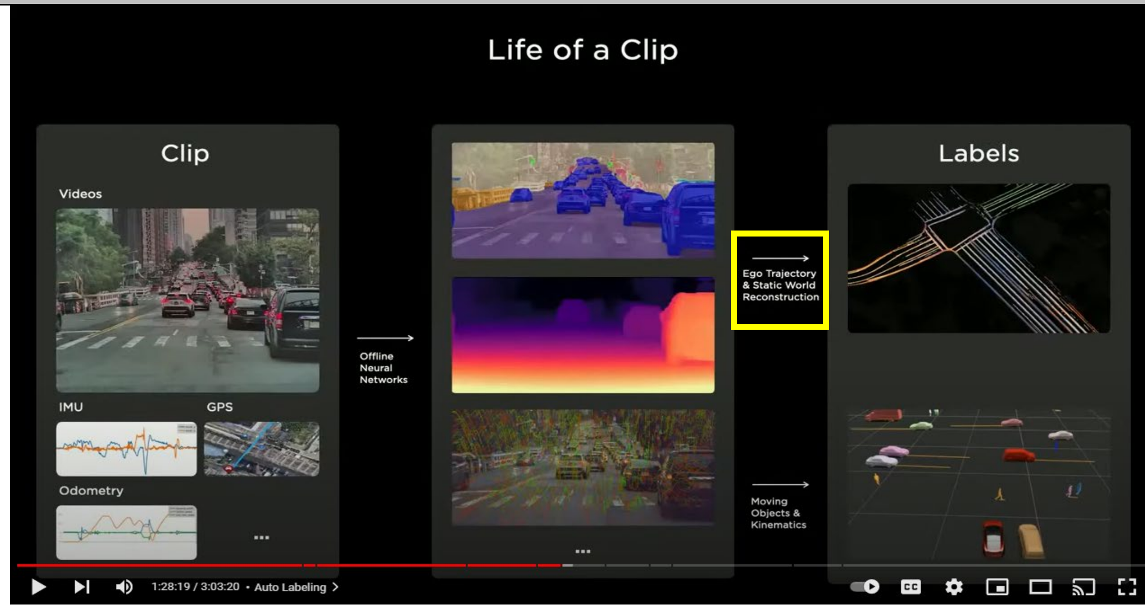
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 Around 1:11:20, the speaker begins describing how Tesla uses a feed of raw images captured by the cameras included in the accused Tesla vehicles, and how the images captured by the cameras pass through a transformer module to re-represent the objects captured in the images taken by the camera into the vector space.

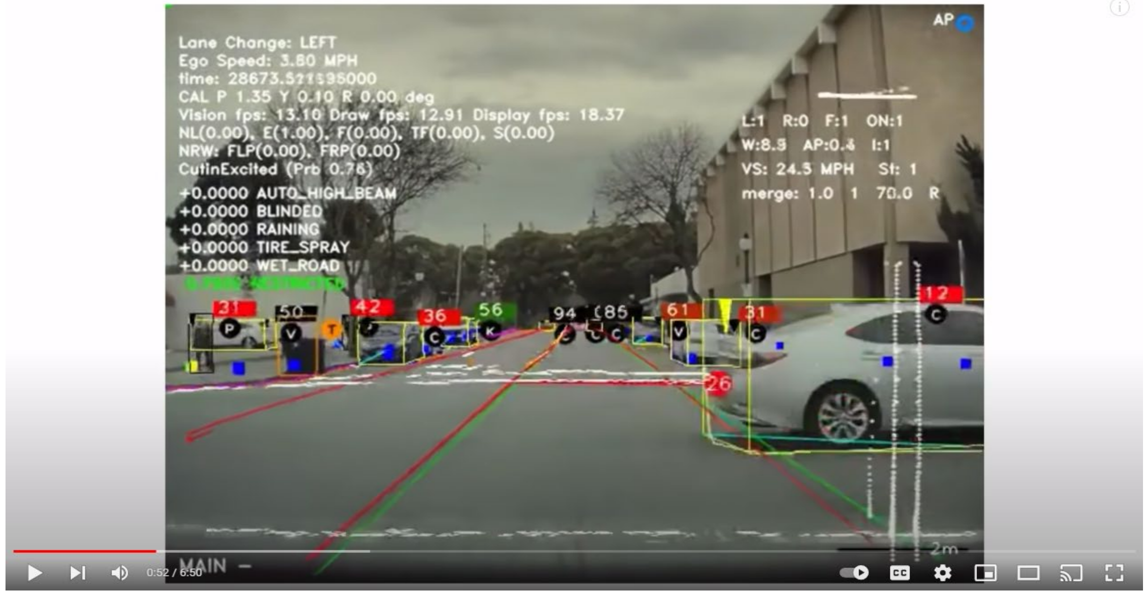
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 At 1:31:23 the speaker describes Tesla's ability to arbitrarily reconstruct 3D static obstacles, and the slide shown in the screenshot above depicts an example of Tesla's reconstructed 3D point cloud. Tesla is able to make this reconstruction from data captured by the cameras and other various sensors included in the accused Tesla vehicles.

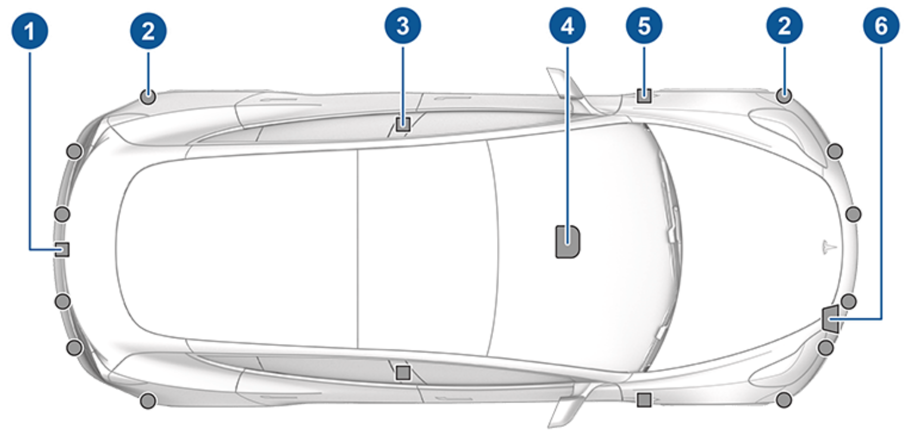
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 At 1:31:58 the speaker is describing Tesla's vector space in more detail, notably touching on how the vector space includes object representations.

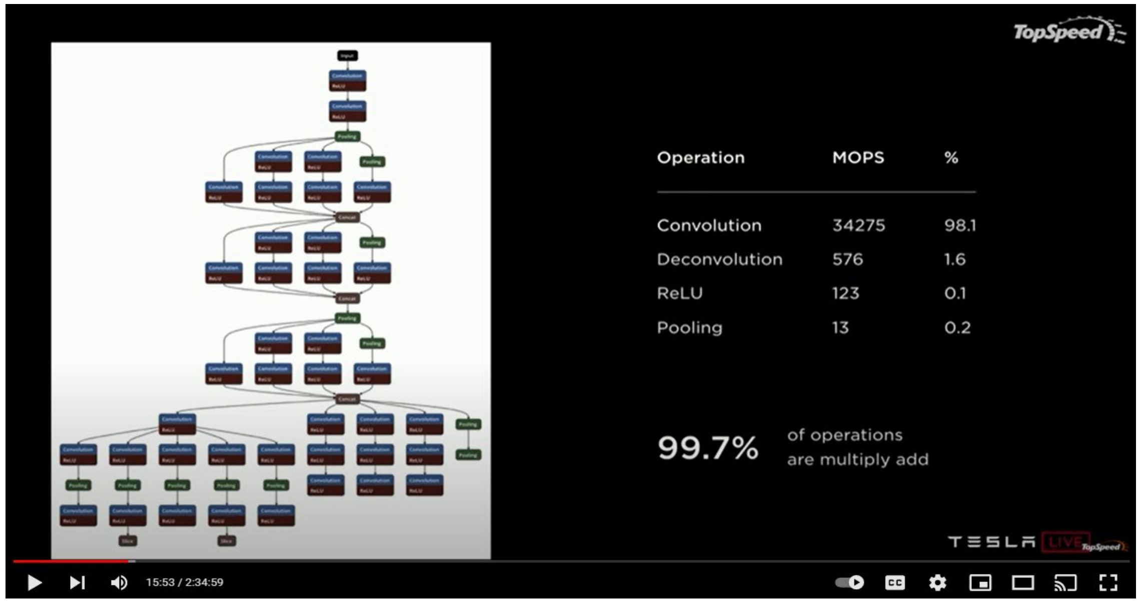
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 At 1:32:48, the speaker explains how Tesla combines everything together in order to produce "amazing" data sets that annotate all of the road texture, all static objects, and all of the moving objects. For additional examples of Tesla's correlating a first circumstance representation with one or more instruction sets, see also, Tesla AI Day 2021(https://www.youtube.com/watch?v=j0z4FweCy4M).

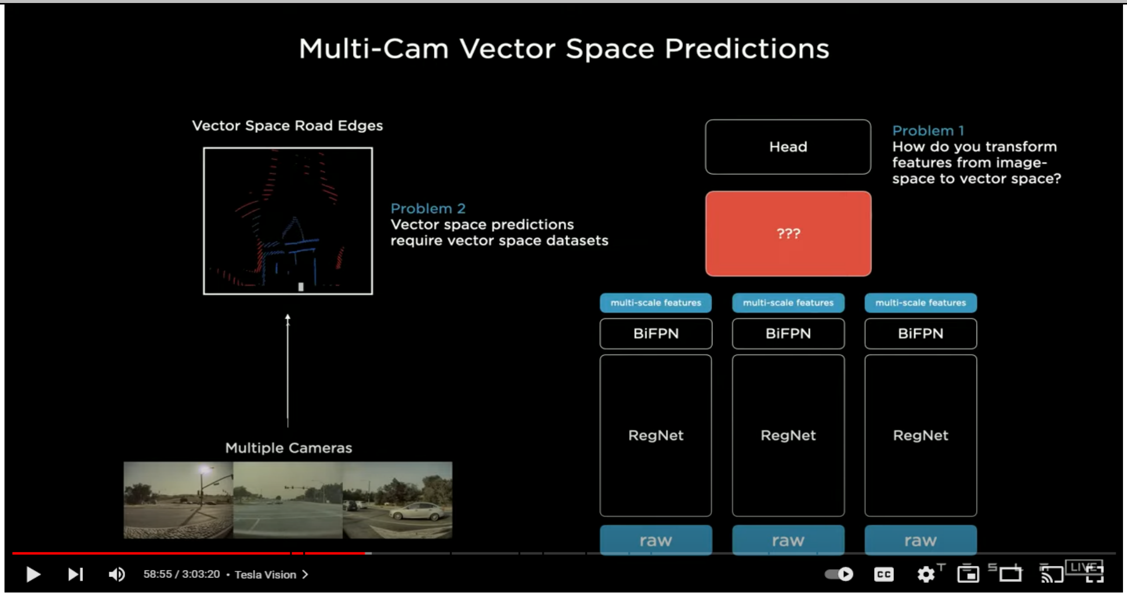
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 At 1:19:43, the speaker describes recording a human's trajectory around a curve, recording/storing the data associated therewith, and sharing that data with the fleet so the accused Tesla vehicles are better able to handle that particular turn and similar turns/curves in the future.

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 At/around 1:28:16, the speaker is describing exemplary labeling of a single clip in which an entity that has dense sensor data like videos, IMU data, GPS, odometry (e.g., speed, steering braking, etc.) are uploaded by Tesla's own engineering cars and/or from customer cars. Note the "Ego Trajectory & Static World Reconstruction" label shown in the screenshot above (yellow annotation). Again, at/around 2:55:29, Mr. Musk describes how all the human drivers are "essential training the neural net as to what is the correct course of action" in any particular circumstance encountered by the accused Tesla vehicles. A further example of Tesla's correlating a first circumstance representation with one or more instruction set is depicted below.

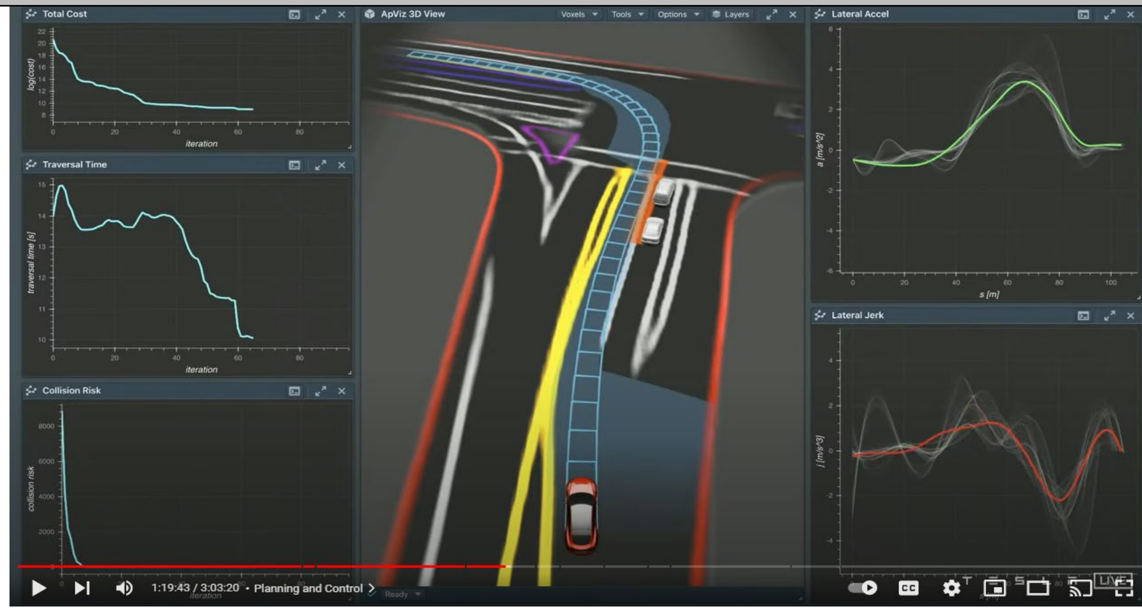
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 At/around 0:50 in the video referenced above/cited below, a myriad of instruction labels considered by the accused Tesla vehicles while in operation can be seen. Examples of such instruction labels include Lane change and Ego speed. Source: https://www.youtube.com/watch?v=zRnSmw1i_DQ A bevy of sensors are included in the accused Tesla vehicles. Examples of sensors included in the accused Tesla vehicles include an array of outward-facing cameras that capture the environment surrounding the vehicle. Optionally, ultrasonic sensors and radar can be equipped.

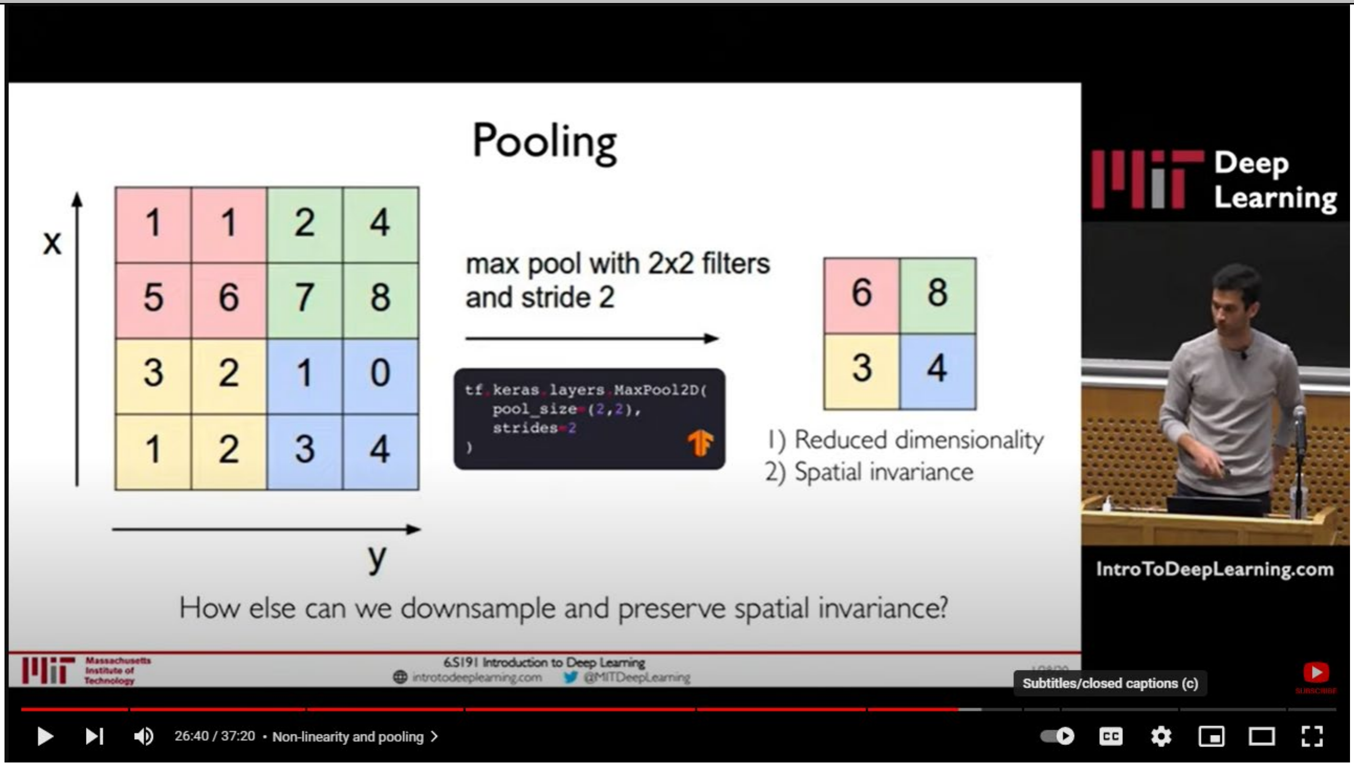
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	TESLA | Model 3 Owner's Manual Model 3 Owner's Manual Overview Opening and Closing Storage Areas Seating and Safety Restraints Connectivity Driving Autopilot **About Autopilot** Traffic-Aware Cruise Control Autosteer Navigate on Autopilot Traffic Light and Stop Sign Control Autopark Summon Smart Summon Active Safety Features Dashcam, Sentry, and Security Climate Navigation and Entertainment Charging and Energy Consumption Maintenance Specifications Roadside Assistance Troubleshooting Consumer Information ## How It Works Your Model 3 includes the following components that actively monitor the surrounding area:  1. A camera is mounted above the rear license plate. 2. Ultrasonic sensors (if equipped) are located in the front and rear bumpers. 3. A camera is mounted in each door pillar. 4. Three cameras are mounted to the windshield above the rear view mirror. 5. A camera is mounted to each front fender. 6. Radar (if equipped) is mounted behind the front bumper. Model 3 is also equipped with high precision electronically-assisted braking and steering systems. In addition, the cabin camera (if equipped) can determine driver inattentiveness and provide alerts when Autopilot is engaged. By default, data from the camera does not leave the vehicle itself. In other words, data is not saved or transmitted unless you enable data sharing. To enable data sharing, touch **Controls > Software > Data Sharing > Allow Cabin Camera Analytics**. Cabin Camera Analytics helps Tesla continue to develop even safer vehicles in the future. See Cabin Camera. Source: https://www.tesla.com/ownersmanual/model3/en_au/GUID-EDA77281-42DC-4618-98A9-CC62378E0EC2.html#:~:text=Ultrasonic%20sensors%20(if%20equipped)%20are,mounted%20to%20each%20front%2 0fender.

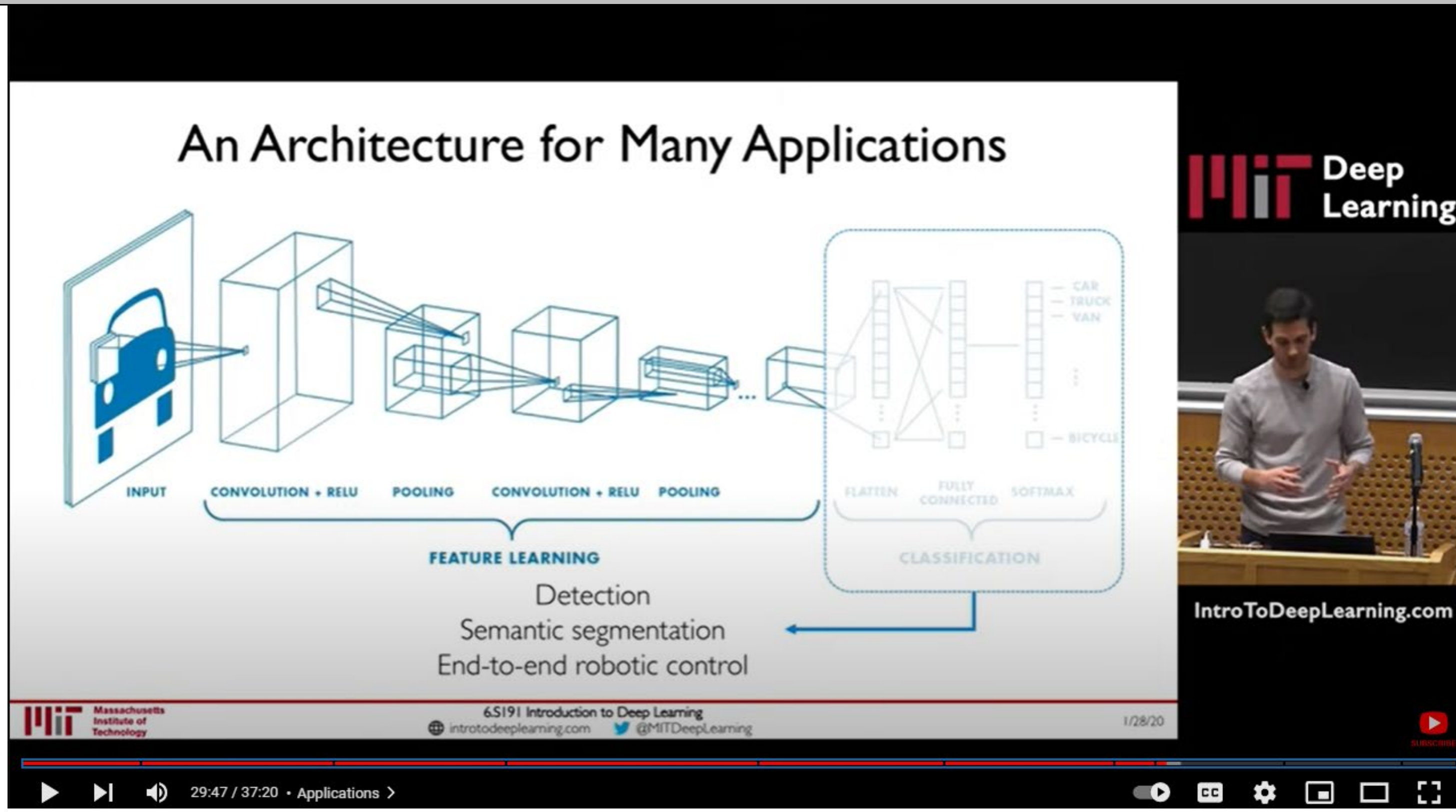
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	Tesla describes how circumstance representations are correlated with various instruction sets for operation of the accused Tesla vehicles during Tesla Autonomy Day 2019 (https://www.youtube.com/watch?v=-b041NXGPZ8) and Tesla AI Day 2021 (https://www.youtube.com/watch?v=j0z4FweCy4M). As examples:  Tesla describes how the accused Tesla vehicles use a convolutional neural network (CNN) to learn and implement autonomous driving. *See* Tesla Autonomy Day 2019 at/around 15:53.

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 Tesla describes how it wants to re-represent the features from image space as vector space features, and that it wants to use vector space predictions from its neural network, which requires vector space data sets, and vector space data sets require vector space labels. *See* Tesla AI Day 2021 at/around 58:31.

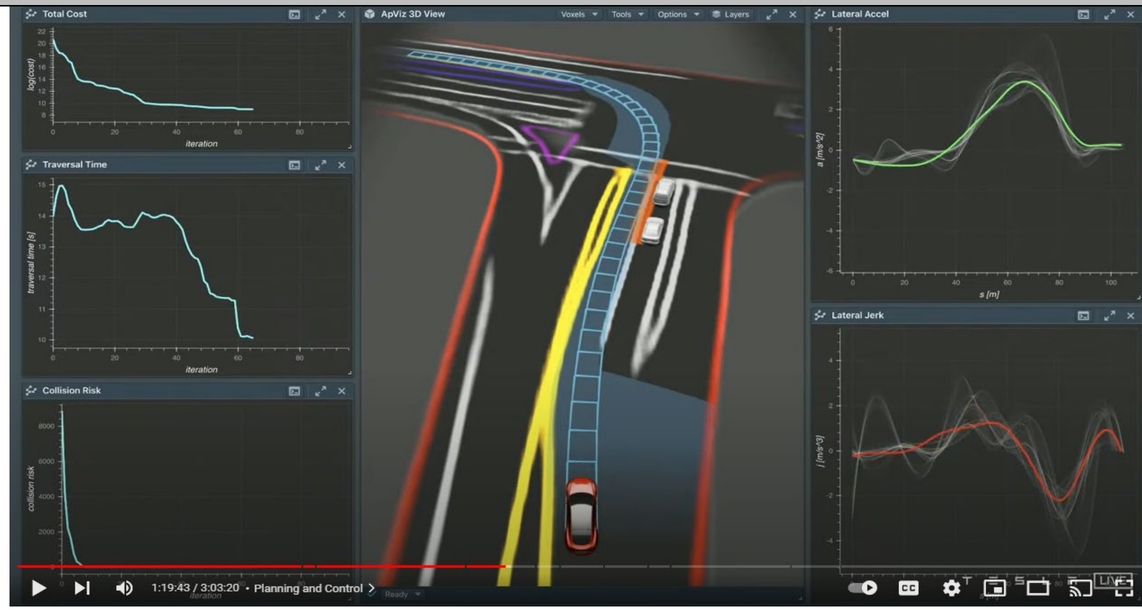
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 Tesla describes how, while driving accused Tesla vehicles, drivers are actually annotating the data captured by the vehicle because the user is steering the wheel, thereby telling the vehicle/Tesla how to navigate/traverse different environment (e.g., circumstance representations representing the environment correlated with driving instructions for traversing the environment). *See* Tesla Autonomy Day 2019 at/around 1:04:10.

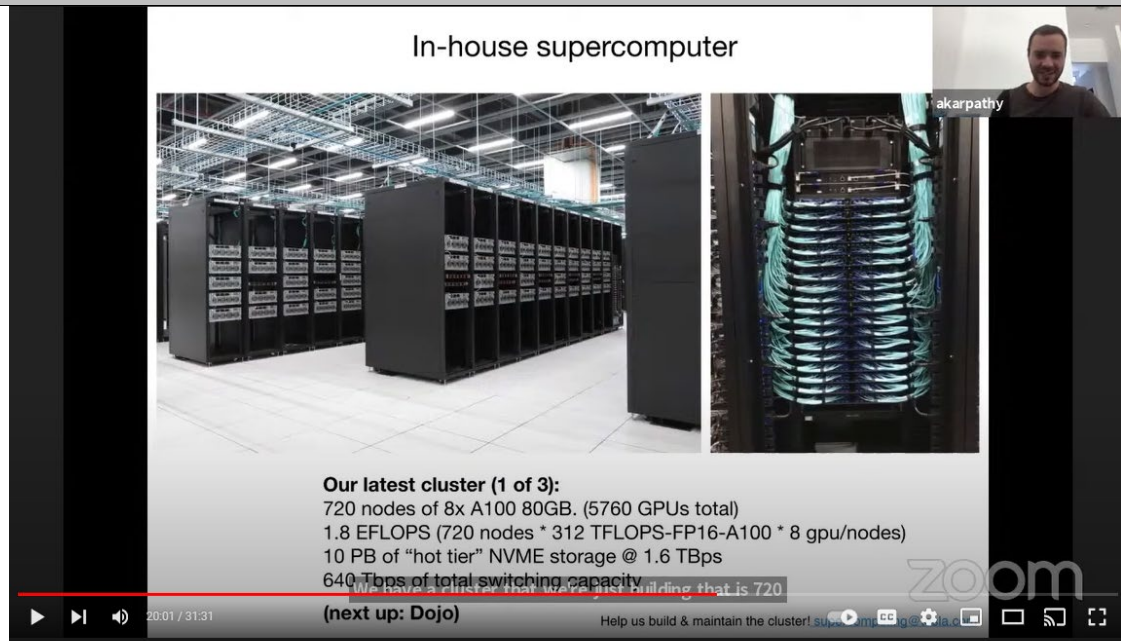
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 Tesla depicts circumstance representation and correlated human trajectory, e.g., trajectory implemented using driving instruction sets at each point in time while the accused Tesla vehicles are taking a turn. *See* Tesla AI Day 2021 at/around 1:19:43. Further, as stated above, circumstance representations include one or more object representations/features. The convolution layers of the CNN in the accused Tesla vehicles store object features correlated via forward connections with output nodes of the CNN, including instructions for driving the vehicle, as described by the expert from MIT speaking in the following video, the "MIT Convolutional Neural Networks" video, found at: https://www.youtube.com/watch?v=iaSUYvmCekI. First, the speaker describes learning features and features being stored/learned in different convolution layers.

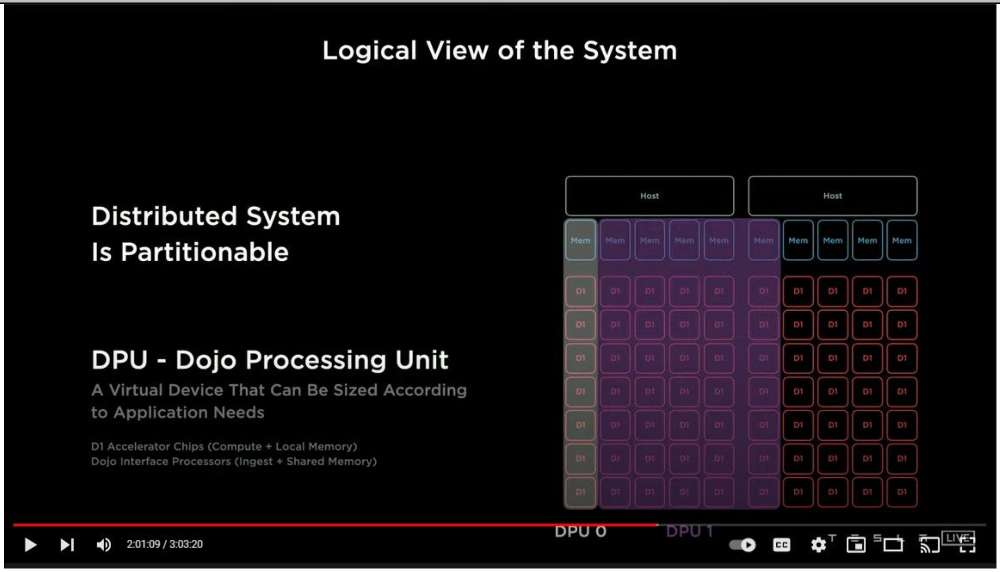
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 MIT Convolutional Neural Networks video at/around 26:40. Next, the speaker describes the overall CNN used for robotic control (e.g., autonomous driving) where features in convolutional layers of the CNN are correlated with control instructions at the end of the CNN via forward connections.

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 *See* MIT Convolutional Neural Network video at/around 29:40. During its Autonomy Day 2019 and AI Day 2021 presentations, Tesla goes on to describe how the instruction sets for operating the device is learned in a learning process that includes operating the first device at least partially by a user. For example:

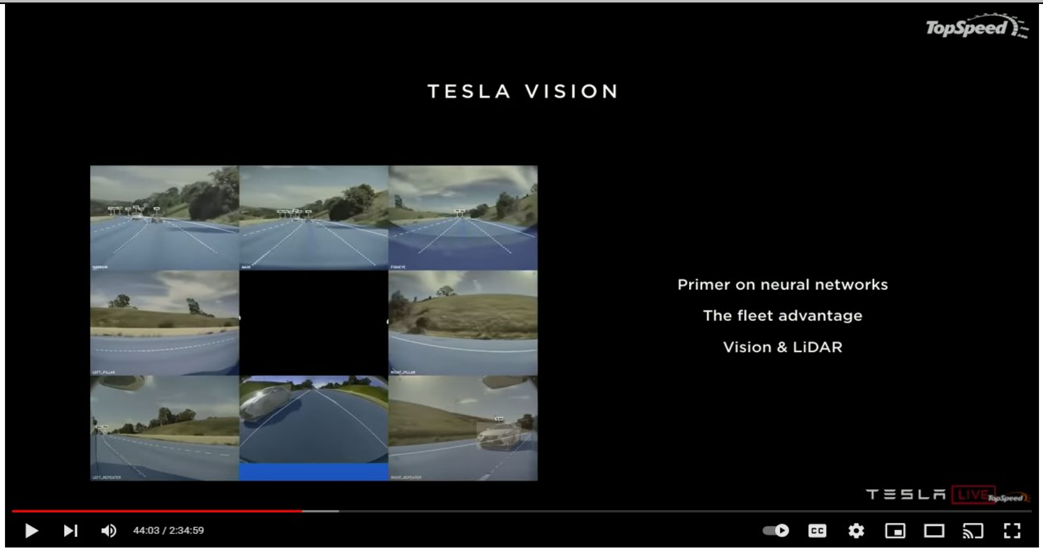
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 At/around 1:04:10 in Tesla's Autonomy Day 2019 presentation, the speaker describes, while driving one of the accused Tesla vehicles, the user is annotating the data because the user is steering the wheel, the user is telling Tesla how to traverse different environments, that one person in the fleet taking, e.g., a left turn through a particular intersection, is providing Tesla with invaluable data corresponding to GPS coordinates, inertial measurements, wheel angle, wheel ticks, etc. thus enabling Tesla to understand the path that particular user took through that particular turn. Tesla then uses this data to supervising/training the CNN (e.g., training through supervised learning). This data is then shared with the entire fleet through the neural network. The speaker goes on to state that Tesla is merely trying to imitate how people drive in the real world. At/around 1:04:01 in Tesla's Autonomy Day 2019 presentation, the speaker describes how Tesla vehicles/Tesla's neural network learns from drivers training the network. At/around1:52:20, it is stated human drivers are monitored (e.g., monitored by a camera), and that the CNN implements autonomous driving based on this learned knowledge.

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 At/around 1:19:43 in Tesla's AI Day 2021 presentation, Tesla describes how the path of the accused Tesla vehicles took a trajectory through the curve shown above that looks substantially similar to humans who had driven through the same turn in accused Tesla vehicles in the past. For example, the supercomputer included in the 474 Accused Products further includes volatile and non-volatile memory, such as NVME or DRAM respectively, for, e.g., storing digital pictures correlated with one or more instruction sets for operating a first object of a first application program, along with other executable, machine-readable code. *See, e.g.*, [CVPR'21 WAD] Keynote - Andrej Karpathy, Tesla, available at https://www.youtube.com/watch?v=g6bOwQdCJrc.

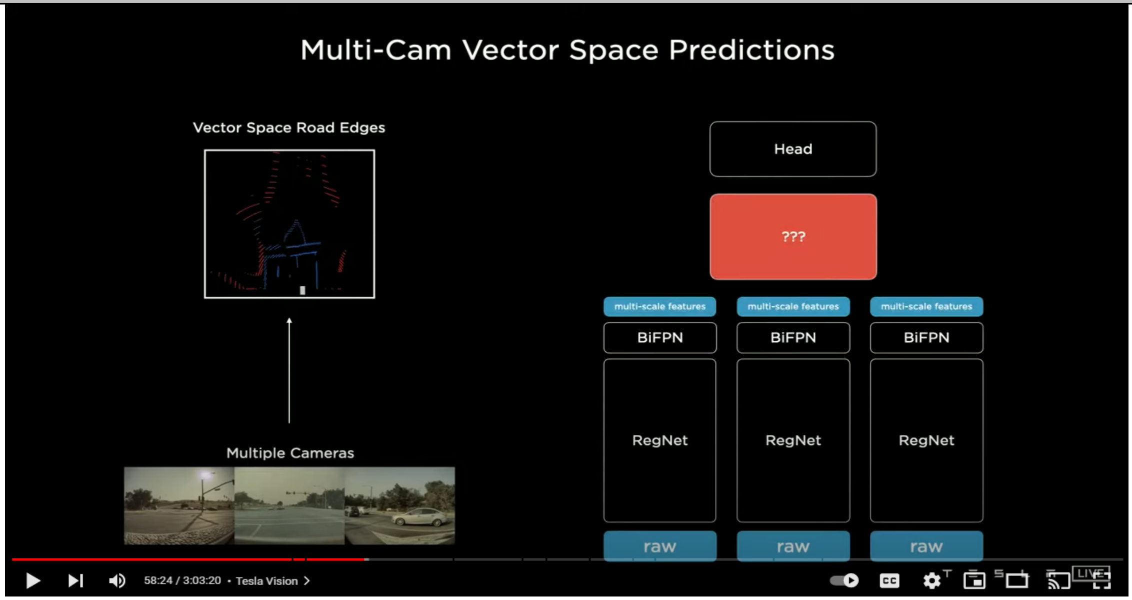
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 *See* Tesla AI Day video, available at https://www.youtube.com/watch?v=j0z4FweCy4M&t=3s *"We do have a major program at Tesla which we don't have enough time to talk about today called "Dojo". That's a super powerful training computer. The goal of Dojo will be to be able to take in vast amounts of data and train at a video level and do unsupervised massive training of vast amounts of video with the Dojo program – or Dojo computer."* Elon Musk hints at Tesla's not-so-secret Dojo AI-training supercomputer capacity – Electrek

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	Karpathy commented on the effort: *"We have a neural net architecture network and we have a data set, a 1.5 petabytes data set that requires a huge amount of computing. So I wanted to give a plug to this insane supercomputer that we are building and using now. For us, computer vision is the bread and butter of what we do and what enables Autopilot. And for that to work really well, we need to master the data from the fleet, and train massive neural nets and experiment a lot. So we invested a lot into the compute. In this case, we have a cluster that we built with 720 nodes of 8x A100 of the 80GB version. So this is a massive supercomputer. I actually think that in terms of flops, it's roughly the number 5 supercomputer in the world."* https://www.inputmag.com/tech/tesla-showed-off-its-massive-supercomputer-for-self-driving- data-processing
[45e]. generating a circumstance representation, wherein the generated circumstance representation represents a circumstance detected at least in part by the one or more sensors;	The '474 Accused Products are devices further comprising one or more non-transitory machine readable media storing machine readable code that, when executed by the one or more processors, causes the one or more processors to perform at least generating a circumstance representation, wherein the generated circumstance representation represents a circumstance detected at least in part by the one or more sensors. For example, each of the accused Tesla Models S, 3, X, and Y is a device. The Tesla fleet includes multiple accused Tesla vehicles, each of which is capable of accounting for circumstances encountered by drivers while out and about, those circumstances being captured by the multiple cameras included therein. Representations of these captured circumstances are then generated. A particular accused Tesla vehicle may capture and represent circumstances on its own, or each of the accused Tesla vehicles can receive circumstances/representations of circumstances from other vehicles in Tesla's fleet, e.g., via Tesla's CNN. At/around 44:00 in Tesla's Autonomy Day 2019, a stream of video from the 8 cameras included in the accused Tesla vehicles can be seen.

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 Further, at/around 41:14 in Tesla's AI Day 2021 presentation, a circumstance representation including object representations representing vehicles, pedestrians, etc. around the accused Tesla vehicle being driven is being shown on the display of the vehicle.

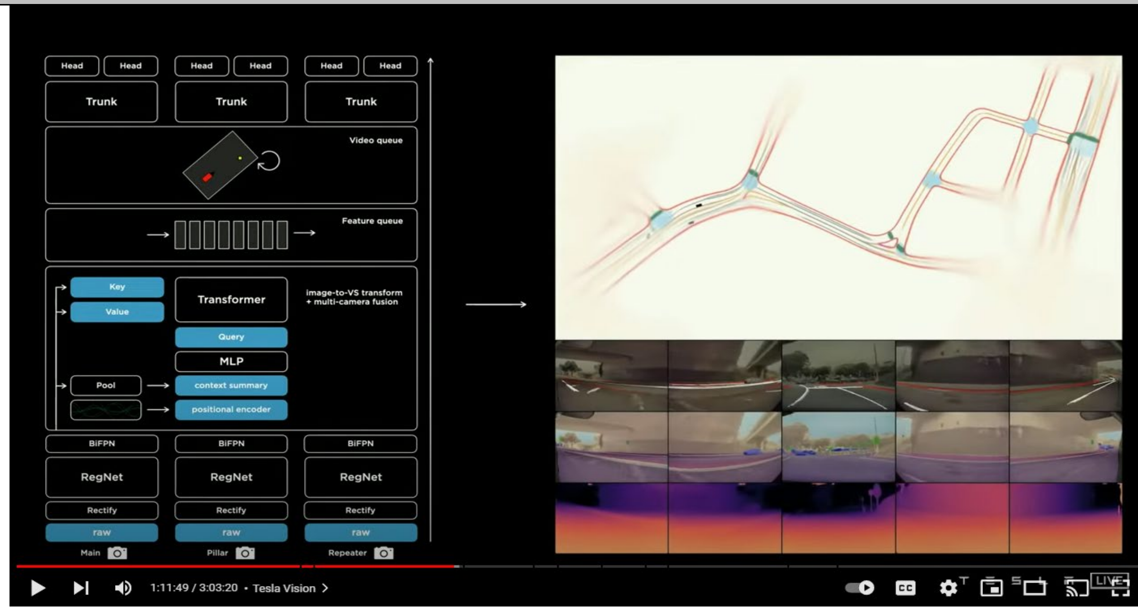
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 At/around 58:10 in Tesla's AI Day 2021 presentation, the presenter mentions Tesla's desire to take all images captured by the cameras and feed them into a single neural network in order to directly output into a vector space

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 Later in Tesla's AI Day 2021 presentation (at/around 1:31:23), the presenter describes how Tesla can arbitrarily reconstruct 3D static objects from data captured by the cameras included in the accused Tesla vehicles.

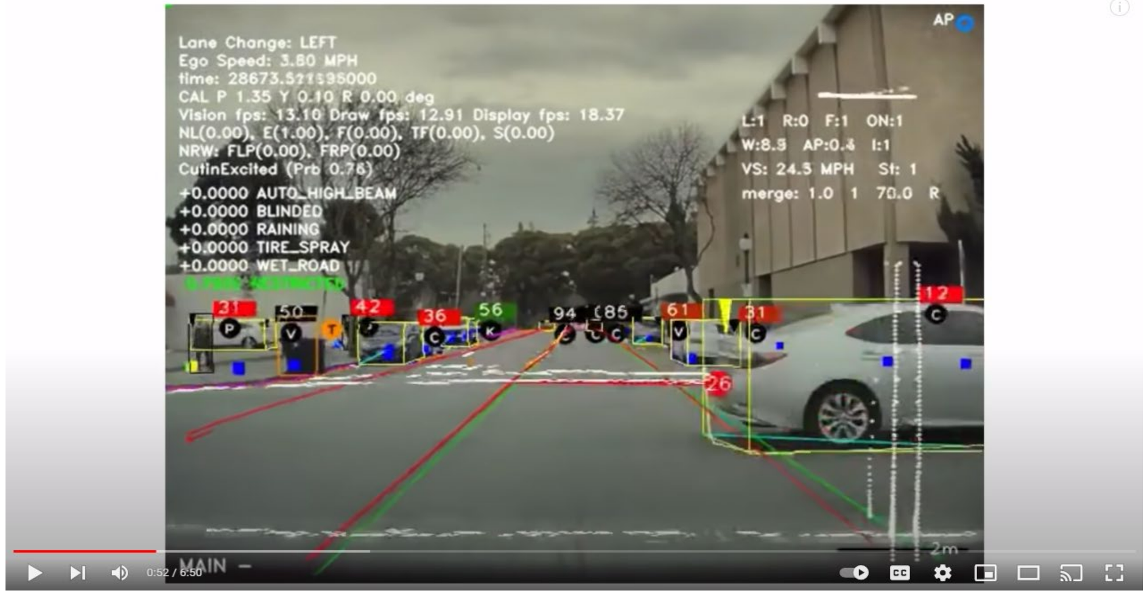
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 Tesla AI Day 2021 goes on to describe: the vector space, including object representations, relied upon by the accused Tesla vehicles (at/around 1:31:58); combining all captured data together to make data sets that annotate all road texture, static objects, and/or moving objects represented in the captured data (at/around 1:32:48); and feeding raw images through a transformer module to re-represent the data in a vector space (at/around 1:11:20).

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	

<table>
<tr><th>U.S. Patent No. 11,663,474</th><th>Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]</th></tr>
<tr><td></td><td>

Circumstance representations, including object representations (e.g., bounding boxes) used to represent objects in the environment of the accused Tesla vehicles can be seen at/around 0:50 of the video "Tesla Autipilot is better than you think! Here's why. (part 1)"</td></tr>
</table>

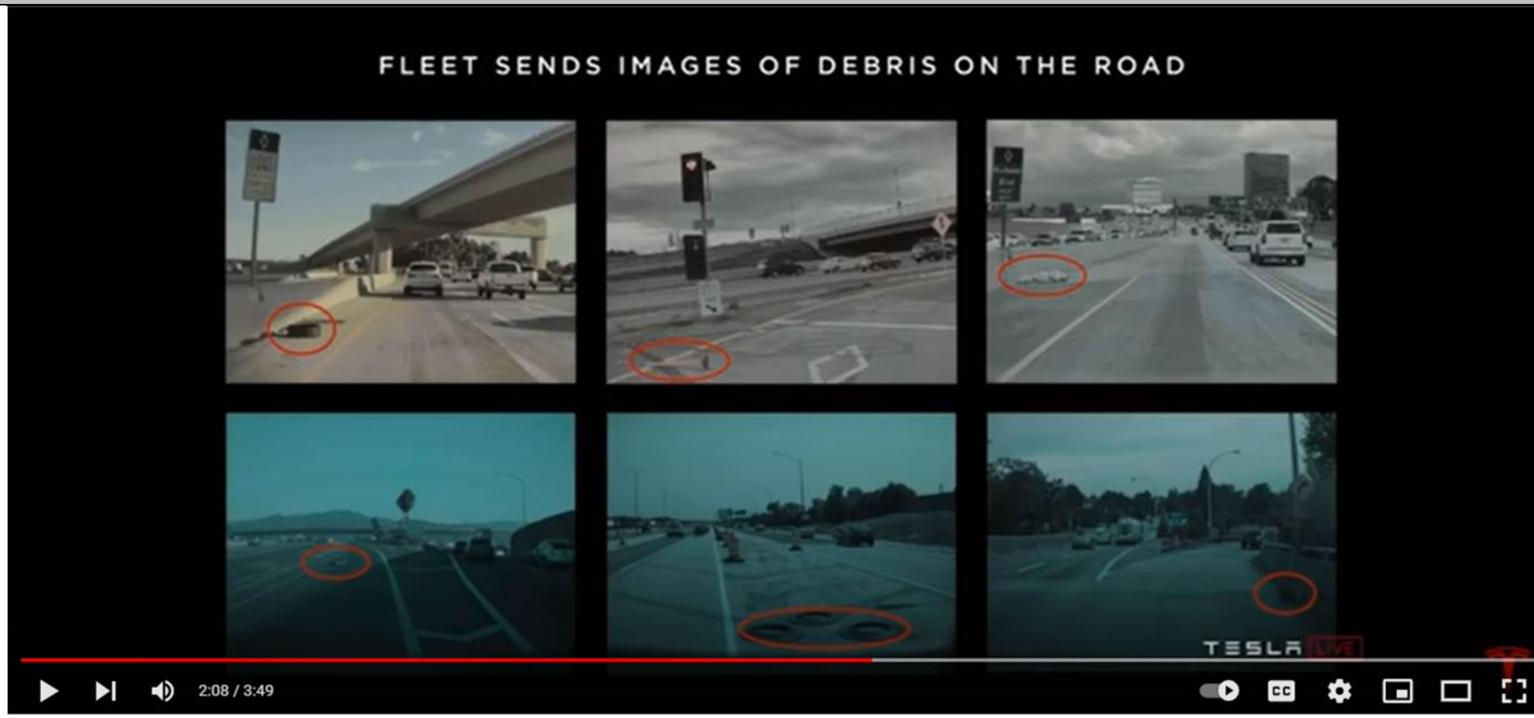
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 Source: https://www.youtube.com/watch?v=zRnSmw1i_DQ Moreover, the 474 Accused Products further generates circumstance representations. According to Tesla, these circumstance representations (or circumstances) are learned while drivers drive Tesla vehicles that are a part of the 474 Accused Products. Examples of circumstances that can be learned include, effectively everything captured/sensed/measured by the sensors (e.g., cameras, ultrasonic, telemetry, etc.) included in each Tesla vehicle in the fleet that is included in the 474 Accused Products. by the driver to navigate those circumstances. This driving knowledge learned by one Tesla vehicle is transmitted to the supercomputer/simulator before being distributed to all vehicles in the Tesla fleet via over-the-air (OTA) software updates. Therefore, the fleet enables each accused Tesla vehicle included therein to autonomously implement driving instructions learned on one or more originating vehicles when similar circumstances are detected by any of the vehicles in Tesla's fleet, thereby enabling Tesla's autonomous driving capabilities.

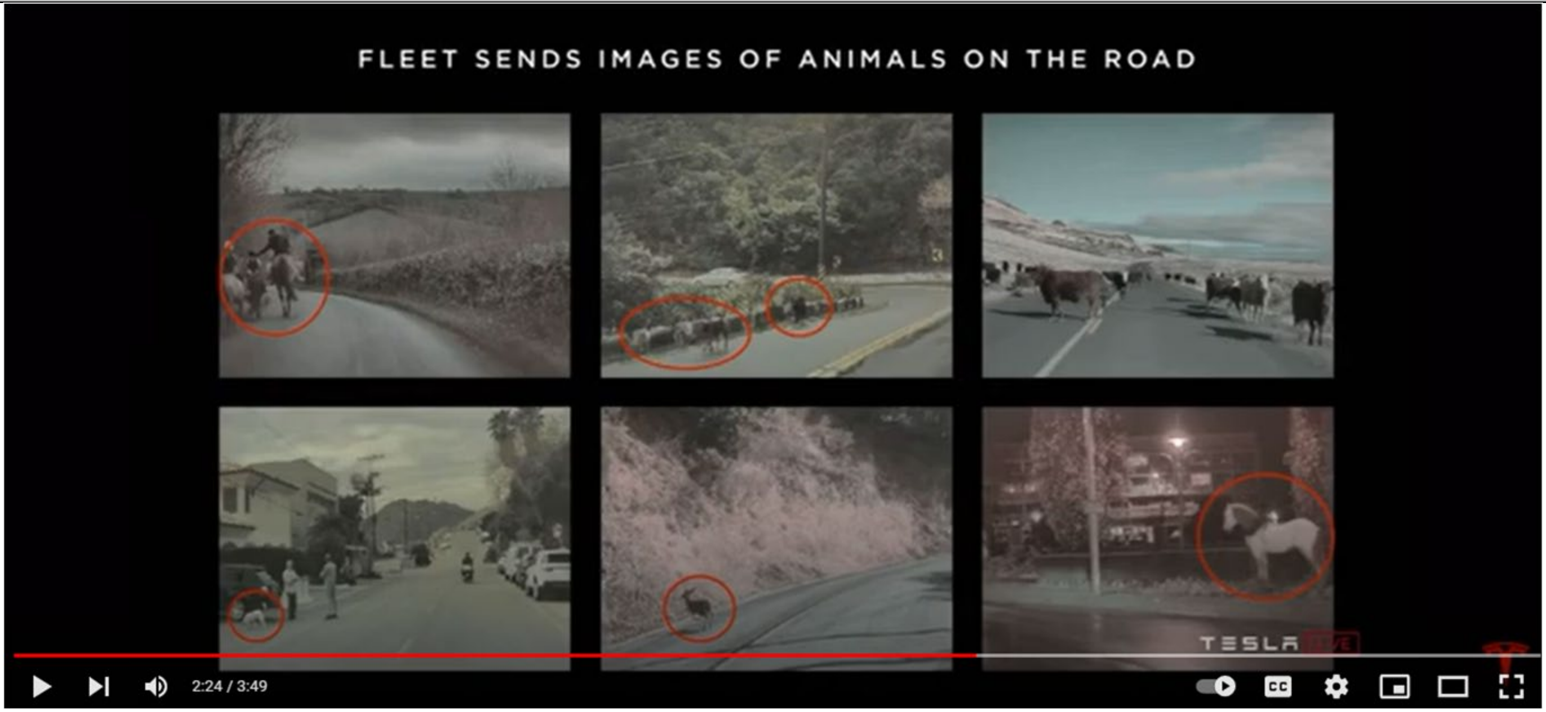
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	Further, to effectuate the circumstance representations, the sensors of [45b] included a Tesla vehicle detect objects. For example, in the previous analysis of driving behind the car with the bike mounted to it, the sensors (e.g., cameras) included in the vehicle detects the bike as an object. Tesla explains object detection as follows:  "so object detection is something we care a lot about we'd like to put bounding boxes around say the cars and the objects here because we need to track them and we need to understand how they might move around so again we might ask human annotators to give us some annotations for these and humans might go in and might tell you that ok those patterns over there are cars and bicycles and so on and you can train your neural network on this, but if you're not careful, the neural network all will make miss predictions in some cases."

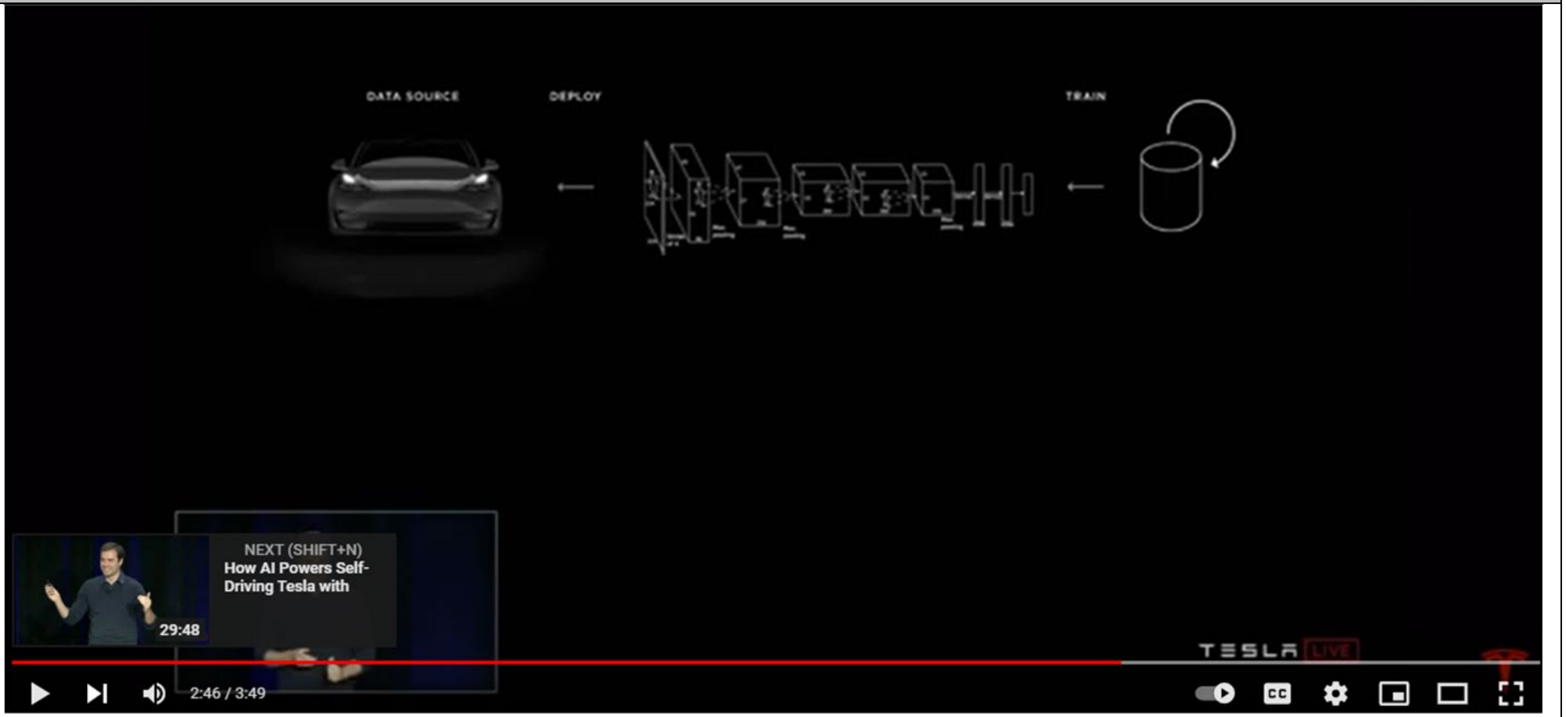
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 "so as an example if we stumble by a car like this that has a bike on the back of it then the neural network actually when I joined would actually create two deductions it would create a car deduction and a bicycle deduction and that's actually kind of correct because I guess both of those objects actually exist but for the purposes of the controller and a planner downstream you really don't want to deal with the fact that this bicycle can go with the car the truth is that that bike is attached to that car so in terms of like just objects on the road there's a single object a single car and so what you'd like to do now is you'd like to just potentially annotate lots of those images as this is just a single car so the process that we that we go through internally in the team is that we take this image or a few images that show this pattern and *we have a mechanism a machine learning mechanism by which we can ask the fleet to source us examples that look like that and the fleet might respond with images that contains those patterns* [*emphasis* added]"

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 "so as an example these six images might come from the fleet they all contain bikes on backs of cars and we would go in and we would annotate all those as just a single car and then the performance of that detector actually improves and the network internally understands that hey when the bike is just attached to the car that's actually just a single car and it can learn that given enough examples and that's how we sort of fix that problem … now the fleet doesn't just respond with bicycles on backs of cars we look for all the thing we look for lots of things all the time."

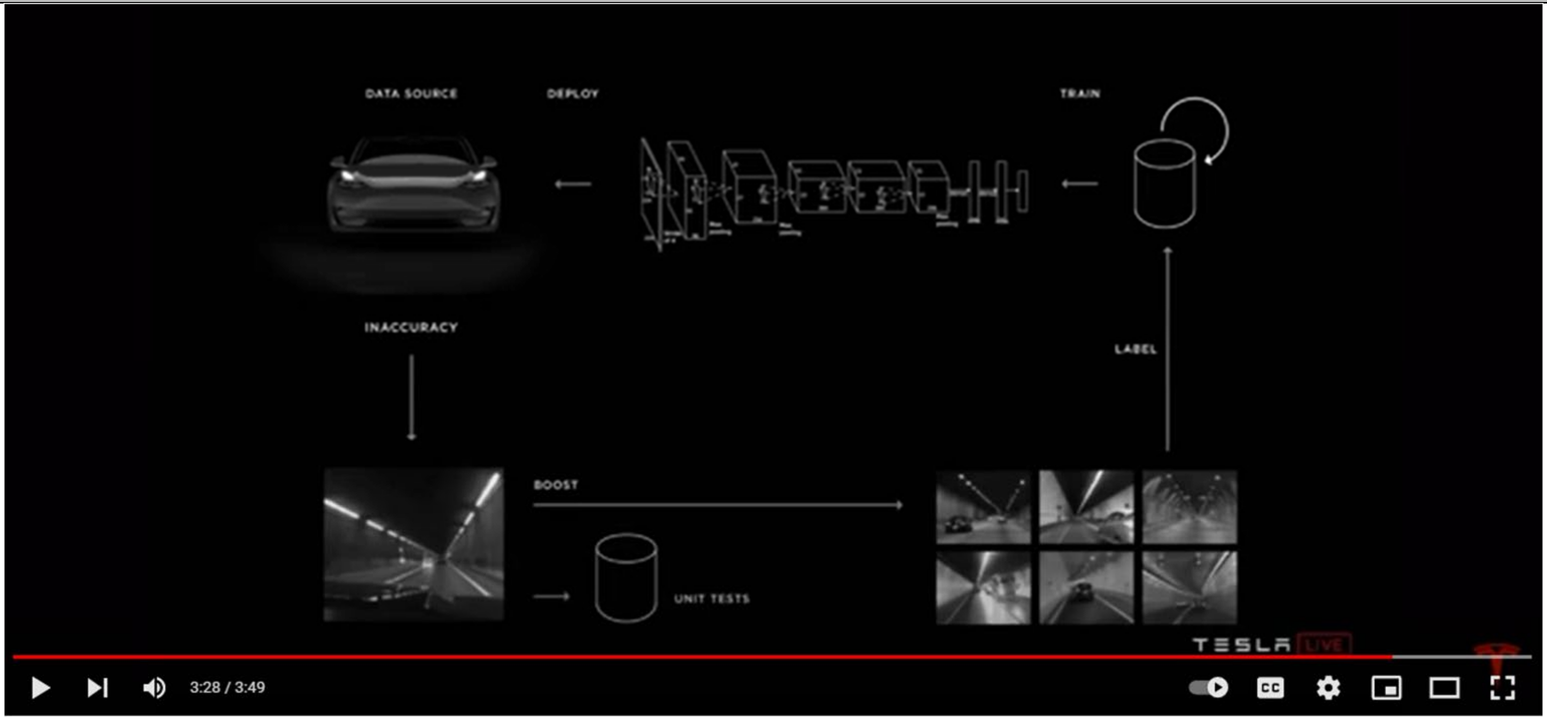
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	  "so for example we look for boats and the fleet can respond with boats we look for construction sites and the fleet can send us lots of construction sites from across the world we look for even slightly more rare cases."

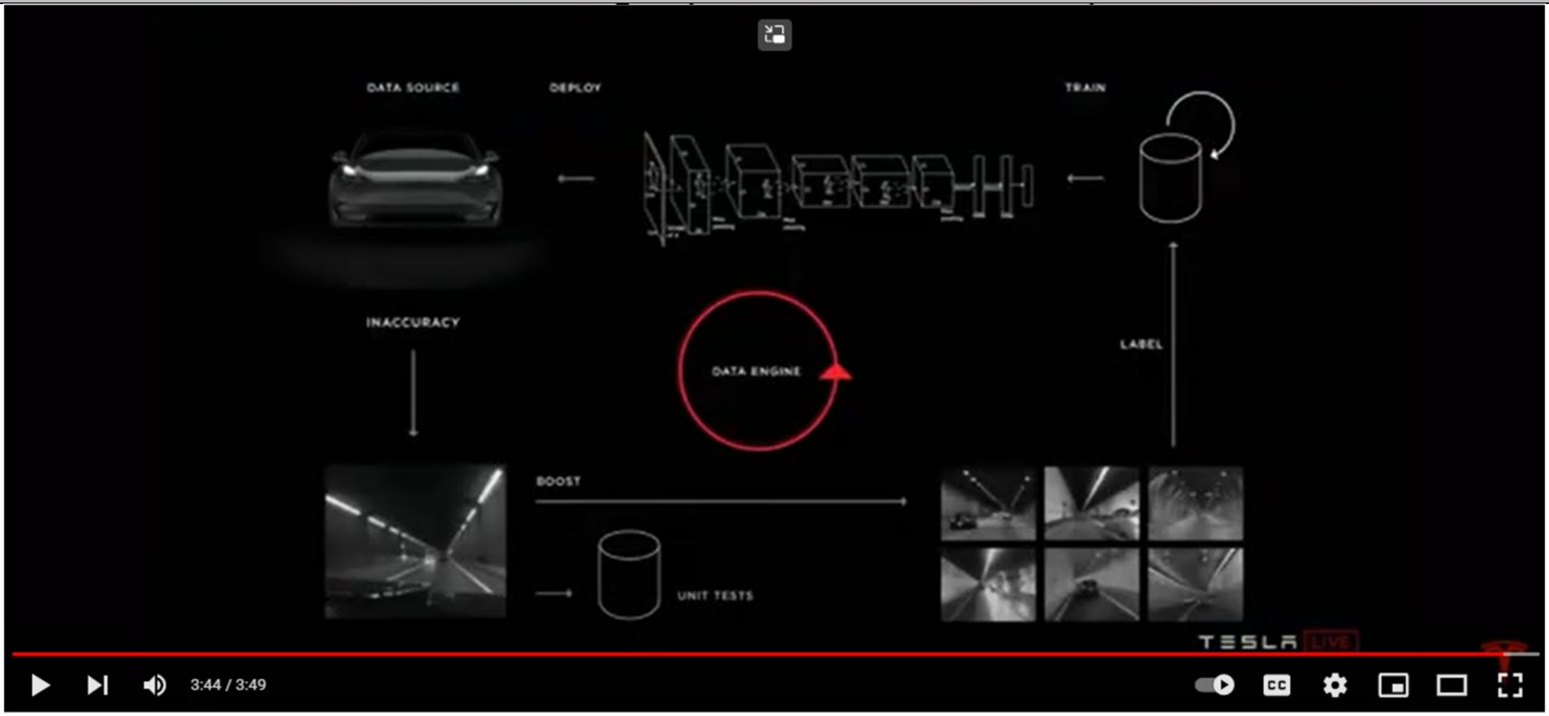
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 "so for example finding debris on the road is pretty important to us so these are examples of images that have streamed to us from the fleet that show tires cones, plastic bags and things like that if we can source these at scale we can annotate them correctly and *the neural network will learn how to deal with them in the world* [*emphasis* added]."

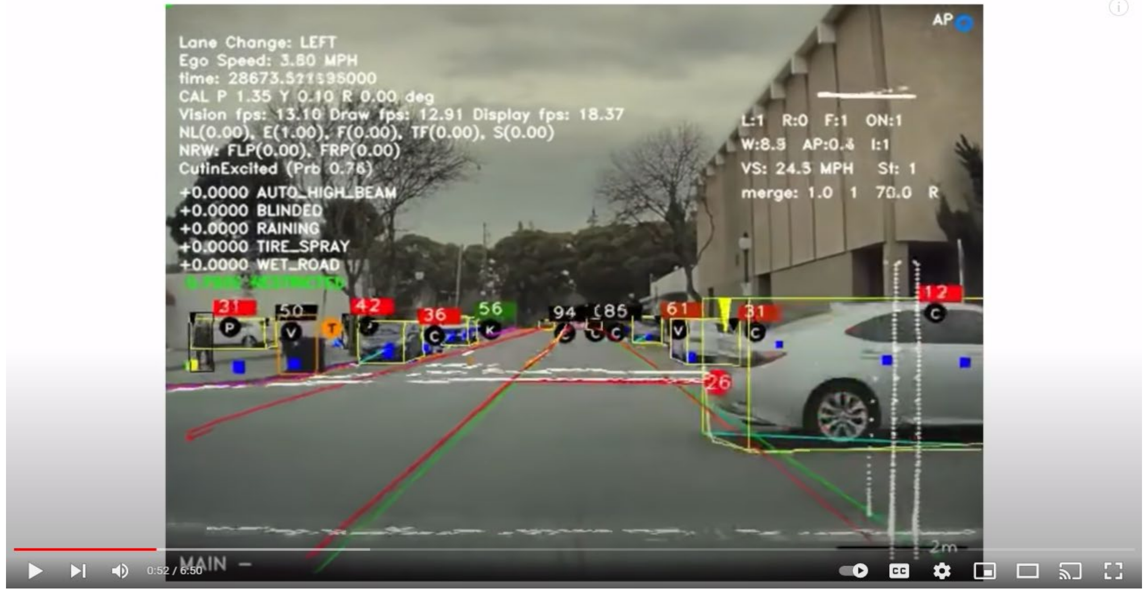
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 "here's another example, animals of course also a very rare occurrence, an event, but we wanted them neural network to really understand what's going on here…that these are animals and we want to deal with that correctly."

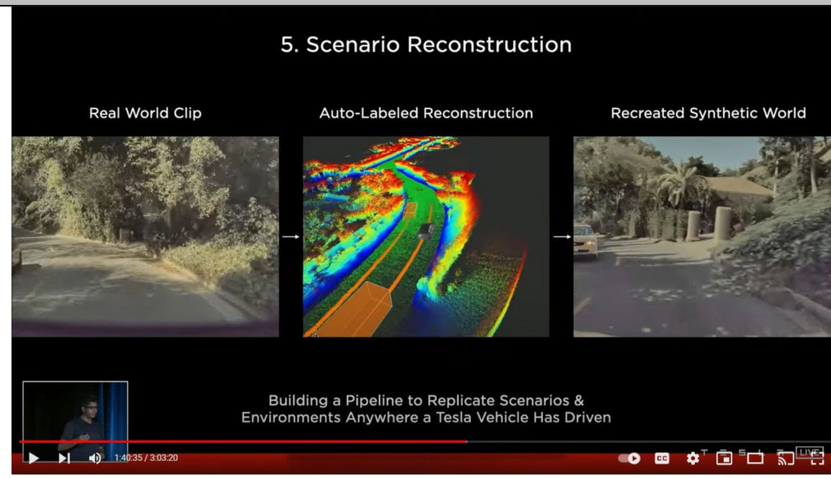
<table>
<tr><th>U.S. Patent No. 11,663,474</th><th>Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]</th></tr>
<tr><td></td><td>

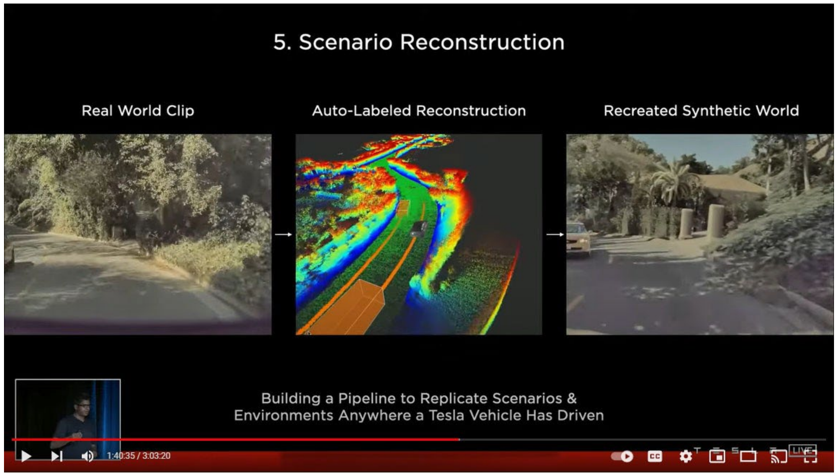
"so to summarize the process by which we iterate on neural network predictions looked something like this – we start with a seed dataset that was potentially sourced at random, we annotate that dataset, and then we train your networks on that dataset and put that in the car, and then we have mechanisms by which we notice inaccuracies in the car when this detector may be misbehaving."</td></tr>
</table>

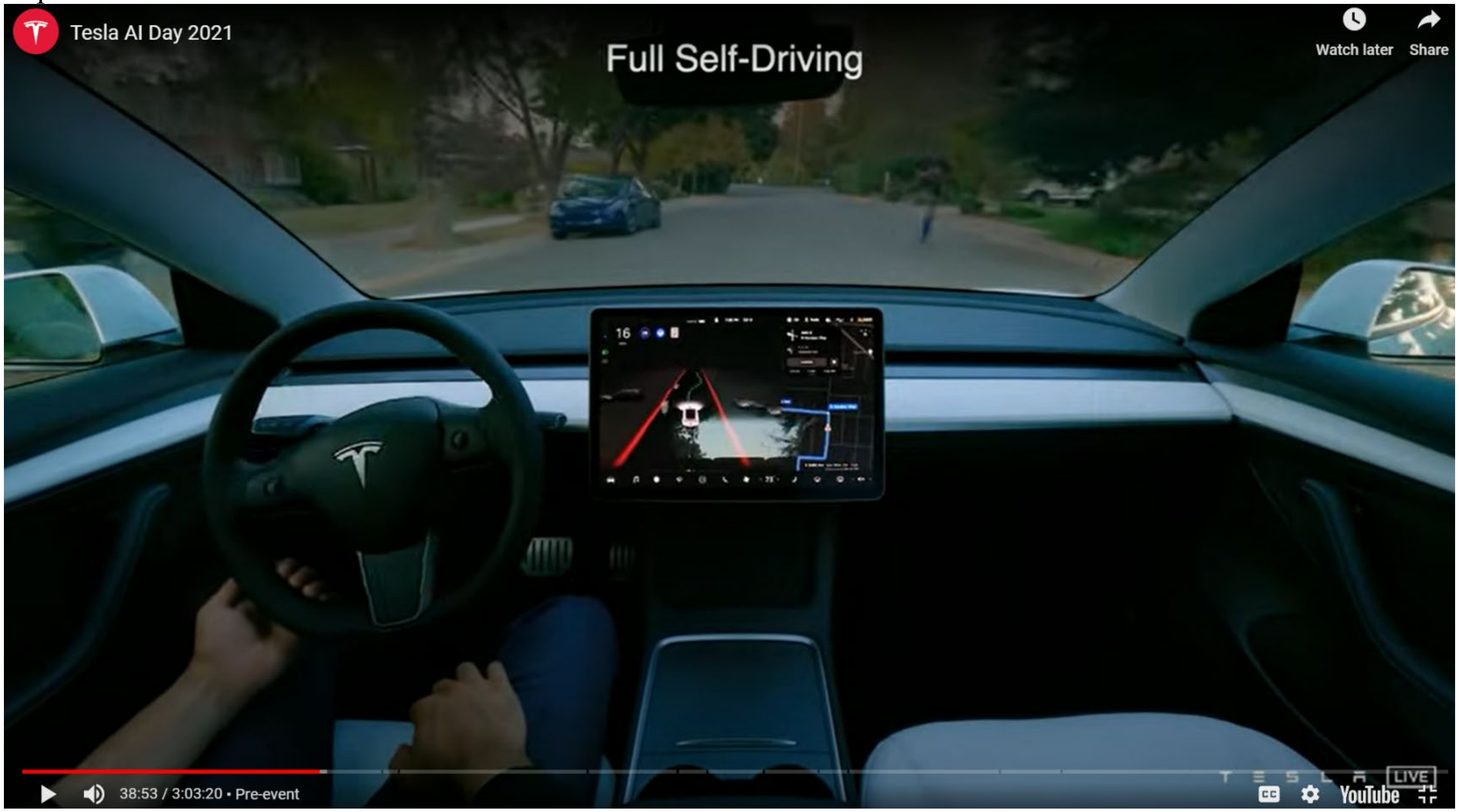
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 "so, for example, if we detect that the neural network might be uncertain or if we detect that or *if there's a driver intervention or any of those settings we can create this trigger infrastructure that sends us data of those inaccuracies and so, for example, if we don't perform very well on lane line detection on tunnels then we can notice that there's a problem in tunnels that imagine would enter our unit tests so we can verify that we've actually fixing the problem over time but now what you do is to fix this inaccuracy you need to source many more examples that look like that so we asked the fleet to please send us many more tunnels and then we label all those tunnels correctly, we incorporate that into the training set and we retrain the network, redeploy, and iterate the cycle over and over again* [*emphasis* added]."

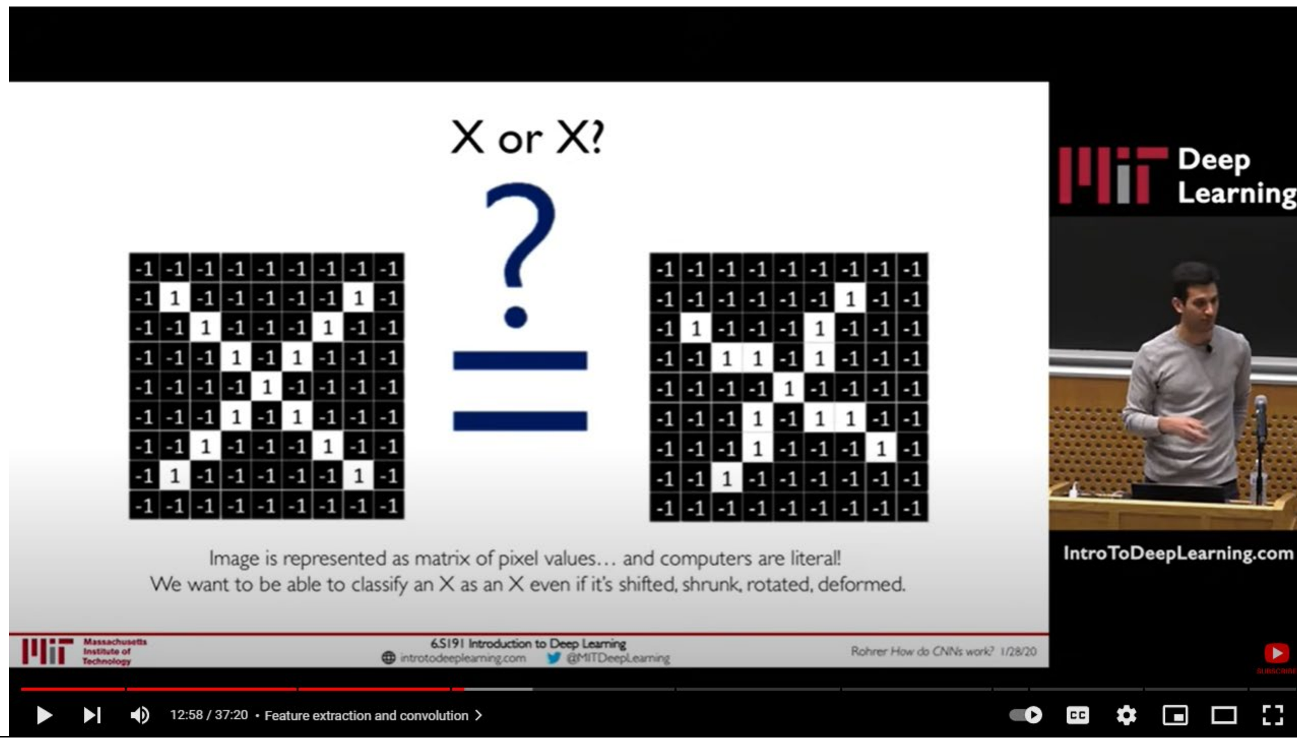
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 "so we refer to this iterative process by which we improve these predictions as the *data engine so iteratively deploying something potentially in shadow mode sourcing inaccuracies and incorporating the training set over and over again and we do this basically for all the predictions of these neural networks* [*emphasis* added]." Source: https://www.youtube.com/watch?v=33K3id2xNAE&t A further example of Tesla's correlating a first circumstance representation with one or more instruction set is depicted below.

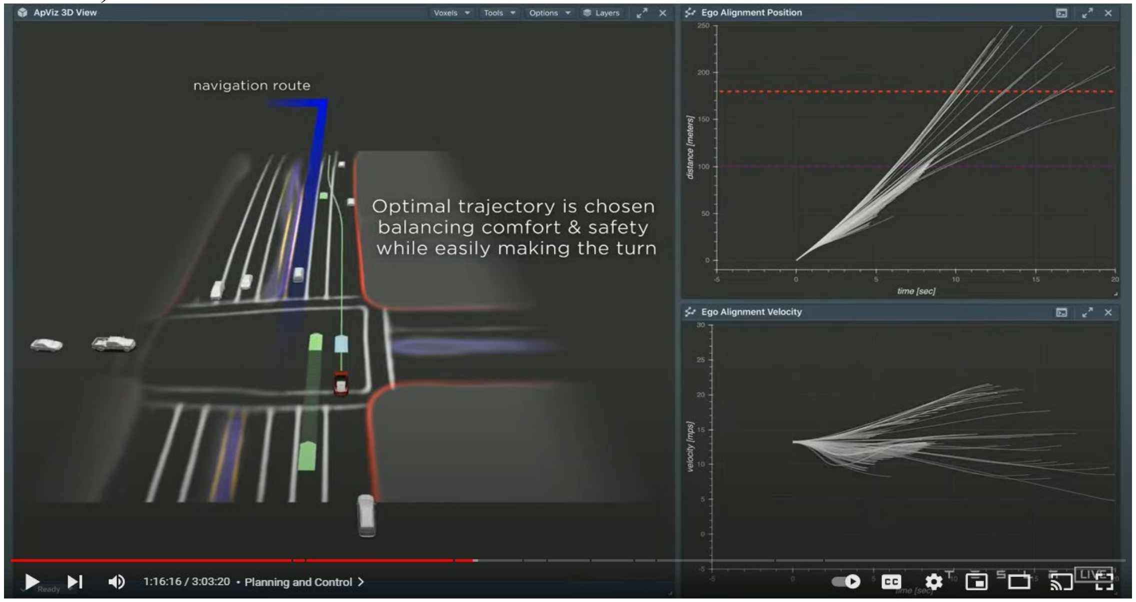
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 At/around 0:50 in the video referenced above/cited below, a myriad of instruction labels considered by the accused Tesla vehicles while in operation can be seen. Examples of such instruction labels include Lane change and Ego speed. Source: https://www.youtube.com/watch?v=zRnSmw1i_DQ Additional examples of Tesla simulating real world driving environments using real world data and generated data can be seen below.

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	 *See* Tesla AI Day video, available at https://www.youtube.com/watch?v=j0z4FweCy4M&t=3s  *Id*.

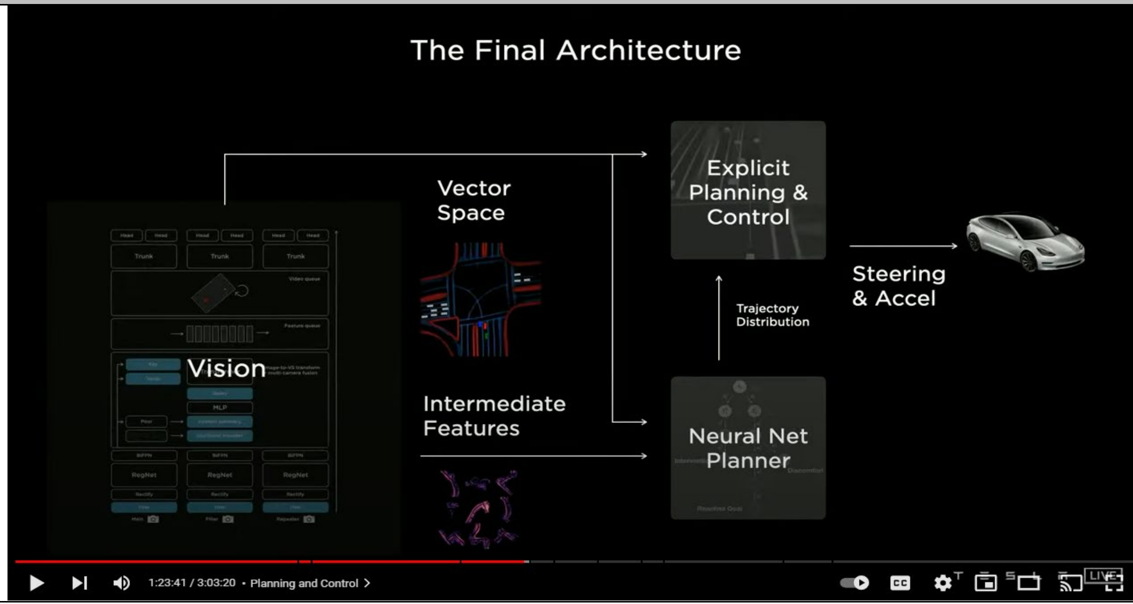
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	The real world and generated data includes video data, which includes object representations. Given that the video data is from the real world and generated simulations of real world environments the object representations represent objects of the application. See, for example, the simulated environment below showing a simulation of a person near the simulated car from the first application program (AI simulation). See Tesla AI Day video, available at https://www.youtube.com/watch?v=j0z4FweCy4M&t=3s As shown in the above-cited video, Tesla's simulation system is capable of behaving appropriately because the simulated car is provided with driving instructions that are correlated with objects, e.g., simulated people, cars, traffic lights, traffic signs, etc.  *See* Tesla AI Day video, available at https://www.youtube.com/watch?v=j0z4FweCy4M&t=3s
[45f]. determining the one or more instruction sets for operating the second device at least by: inputting	The '474 Accused Products are devices further comprising one or more non-transitory machine readable media storing machine readable code that, when executed by the one or more processors, causes the one or more processors to perform at least determining the one or more instruction sets for operating the second device at least by: inputting at least a portion of the generated circumstance representation into the one or more inputs, and using a correlation between the one or more inputs and the one or more instruction sets for operating the second device

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
at least a portion of the generated circumstance representation into the one or more inputs, and using a correlation between the one or more inputs and the one or more instruction sets for operating the second device; and	For example, the 474 Accused Products determines an instruction set (e.g., steer to the left) for operating vehicles in the Tesla fleet. These instructions are provided to Tesla vehicles based on the circumstance representations gathered by Tesla fleet vehicles, that are correlated to instruction sets as described above, over-the-air. This correlation may be based on one similarity between one or more object representations detected/present in the circumstance representation and the first one or more object representations. https://electrek.co/2021/08/19/watch-tesla-ai-day-livestream-important-news/ at/around 38:45 to 39:00  Further, vehicles in Tesla's fleet can learn from each other. The ability of the fleet to learn from itself is mentioned as a distinct competitive advantage of Tesla by both Mr. Musk and Mr. Karpathy. *See*, for example, Tesla Autonomy Day

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	2019 at/around 38:58, "I think a very powerful sustainable advantage for us is the fleet." At/around 44:40, it is stated that, "it is such a big deal that we have the fleet," and at 55:55, it is reiterated that, "Tesla is in such a unique and interesting position" because of the fleet. Finally, at/around 1:22:14, it is proclaimed that it "[i]s extremely difficult to catch up when Tesla has 100 times more miles per day than everyone else combined," which is a direct result of the fleet. Similarly, while operating in autonomous driving mode, the CNN in the accused Tesla vehicles carries out a comparison between features of objects in a previously learned circumstance representation stored in its convolution layers and features of objects in an incoming circumstance representation. *See* MIT Convolutional Neural Networks at/around 12:58. 

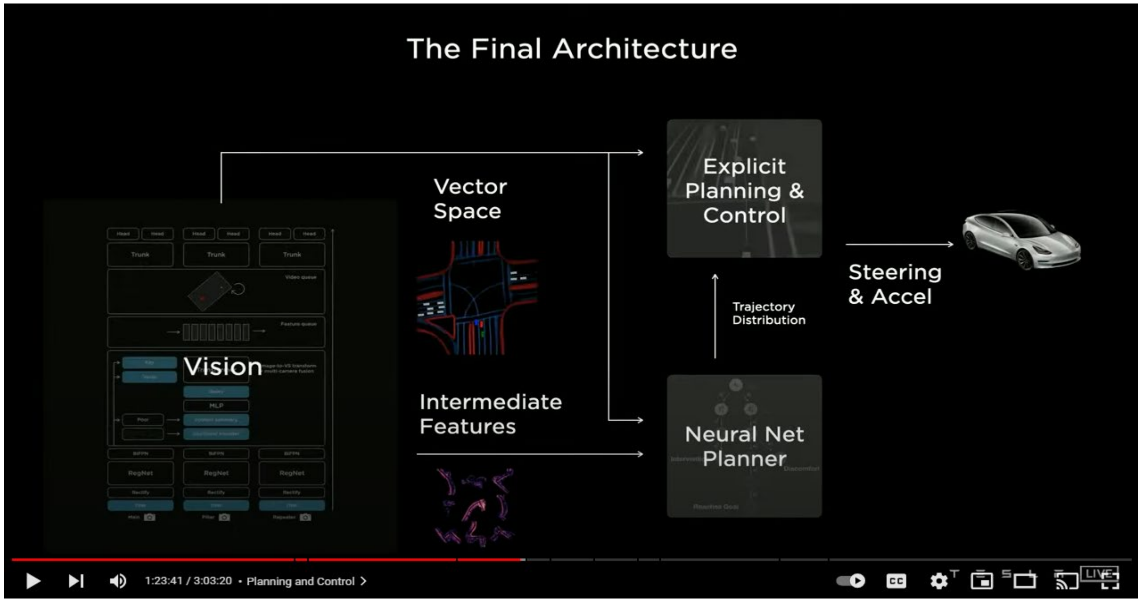
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	Source: https://www.youtube.com/watch?v=iaSUYvmCekI. Further, Tesla's AI Day 2021 presentation (https://www.youtube.com/watch?v=j0z4FweCy4M) describes/presents the following: An example of how searching for previously learned driving instructions for effecting a trajectory correlated with a previously learned circumstance representation that at least partially matches an incoming circumstance representation operates (at/around 1:15:26).  The screenshot above depicts an accused Tesla vehicle attempting to do a lane change. In this instance, the car needs to get over two lanes before making the left-hand turn. To perform this maneuver (getting over two lanes, then making a 90-degree left turn), the accused Tesla vehicle searches over different maneuvers available on the network shared by the Tesla fleet of vehicles (e.g., the CNN and/or simulator). The car tries to pick the safest maneuvers that feel comfortable/natural to the driver, based on data already stored on the network, which itself is based on data observed by

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	the accused Tesla vehicles of the drivers of said vehicles. In some instances, thousands of maneuvers are searched through in real-time, in a very short time span. At/around 2:32:42, the presenter at Tesla's AI Day 2021 describes planning. In essence, Tesla bakes in a search for a previously learned driving instruction(s) effecting a trajectory correlated with a previously learned circumstance representation that, at least partially, matches the incoming circumstance representation being captured by the vehicle. Via network optimization, a plan is devised (e.g., when/how the car will change lanes to the left the first time, when/how the car will change lanes to the left the second time, when/how the car will make the 90-degree left turn). This plan is devised rather quickly. At/around 1:23:40, the presenter at Tesla's AI Day 2021 describes the architecture and the vision system that is implemented in the accused Tesla vehicles. The vision system is described as condensing down video into a vector space (e.g., incoming circumstance representations representing objects in the Tesla's current environment). This video is also consumed by the CNN that stores previously learned circumstance representations correlated with previously learned driving instructions, and that searches for at least partially matching circumstance representation(s) and correlated driving instructions, along with a planner. A trajectory distribution is produced. The trajectory distribution can be optimized accounting for explicit cost functions, human intervention, and other imitation data. The trajectory distribution is then worked into a planning function that causes the car to perform the actions deemed most appropriate based on the comparison of data, and produces the final driving commands (e.g., the commands that control steering, speed, acceleration, turn signals, etc.).

<table>
<tr><th>U.S. Patent No. 11,663,474</th><th>Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]</th></tr>
<tr><td></td><td>
</td></tr>
<tr><td>[45g]. at least in response to the determining, executing the one or more instruction sets for operating the second device, wherein the first device autonomously performs one or more operations defined by the one or more instruction</td><td>The '474 Accused Products are devices further comprising one or more non-transitory machine readable media storing machine readable code that, when executed by the one or more processors, causes the one or more processors to perform, at least in response to the determining, executing the one or more instruction sets for operating the second device, wherein the first device autonomously performs one or more operations defined by the one or more instruction sets for operating the second device.

For example, see the evidence presented in support of [45f] above.

As described by Mr. Musk in Tesla's Autonomy Day 2019 presentation, the Tesla system includes using driving instructions learned from/by one Tesla vehicle to effect the semi/fully-autonomous driving of another Tesla vehicle in the fleet. Mr. Musk goes on to explain that this knowledge is distributed over the Tesla network via over-the-air (OTA) software updates, thereby enabling second, third, fourth, fifth, etc. Tesla vehicles to implement autonomous driving knowledge/instructions learned from the first vehicle.</td></tr>
</table>

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
sets for operating the second device.	The ability of the fleet to learn from itself is mentioned as a distinct competitive advantage of Tesla by both Mr. Musk and Mr. Karpathy. *See*, for example, Tesla Autonomy Day 2019 at/around 38:58, "I think a very powerful sustainable advantage for us is the fleet." At/around 44:40, it is stated that, "it is such a big deal that we have the fleet," and at 55:55, it is reiterated that, "Tesla is in such a unique and interesting position" because of the fleet. Finally, at/around 1:22:14, it is proclaimed that it "[i]s extremely difficult to catch up when Tesla has 100 times more miles per day than everyone else combined," which is a direct result of the fleet. Source: https://www.youtube.com/watch?v=-b041NXGPZ8 (Tesla Autonomy Day 2019 at the various timestamps listed above). During Tesla's AI Day 2021 presentation, Tesla describes what the final architecture of the Tesla/fleet/Software 9.0+/CNN/simulator system is going to look like, and states that it will produce the final steering and acceleration commands for the cars.  Source: https://www.youtube.com/watch?v=j0z4FweCy4M at/around 1:23:40

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
46. The first device of claim 45, wherein the one or more inputs for inputting the at least the portion of the circumstance representation include one or more inputs for inputting at least a portion of one or more object representations, and wherein the generated circumstance representation includes one or more object representations, and wherein the first device is a first vehicle, and wherein the second device is a second vehicle.	The '474 Accused Products comprise the first device of claim 45,wherein the one or more inputs for inputting the at least the portion of the circumstance representation include one or more inputs for inputting at least a portion of one or more object representations, and wherein the generated circumstance representation includes one or more object representations, and wherein the first device is a first vehicle, and wherein the second device is a second vehicle. For example, *see* the evidence presented in support of claims [45a] – [45f] above. The first and second devices are Tesla vehicles, for example. And the imput(s) include input(s) for imputing at least a portion of objection representation(s). Similarly, the generated circumstance representation includes object representation(s).
47. The first device of claim 46, wherein at least an information related	The '474 Accused Products comprise the first device of claim 46, wherein at least an information related to the one or more instruction sets for operating the second device is learned in a learning process that includes operating the second device at least partially by a user.

U.S. Patent No. 11,663,474	**Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]**
to the one or more instruction sets for operating the second device is learned in a learning process that includes operating the second device at least partially by a user.	For example, *see* the evidence provided in claim 45 (particularly [45g]) above. For instance the user may operate the second Telsa vehicle during the learning process.
51. The first device of claim 46, wherein at least a portion of the one or more instruction sets for operating the second device is learned in a learning process that includes operating the second device at least partially by a user.	The '474 Accused Products comprise the first device of claim 46, wherein at least a portion of the one or more instruction sets for operating the second device is learned in a learning process that includes operating the second device at least partially by a user. For example, *see* the evidence presented in claim 45 and 46 above. In this example, at least a portion of the one or more instruction sets for operating the second Tesla vehicle is learned in a learning process that includes operating the second Tesla vehicle at least partially by a driver of that vehicle.
52. The first device of claim 46, wherein the one or more instruction sets for operating the second device	The '474 Accused Products comprise the first device of claim 46, wherein the one or more instruction sets for operating the second device are applied to the first device. For example, vehicles in Tesla's fleet can learn from each other. The ability of the fleet to learn from itself is mentioned as a distinct competitive advantage of Tesla by both Mr. Musk and Mr. Karpathy. *See*, for example, Tesla Autonomy Day 2019 at/around 38:58, "I think a very powerful sustainable advantage for us is the fleet." At/around 44:40, it is

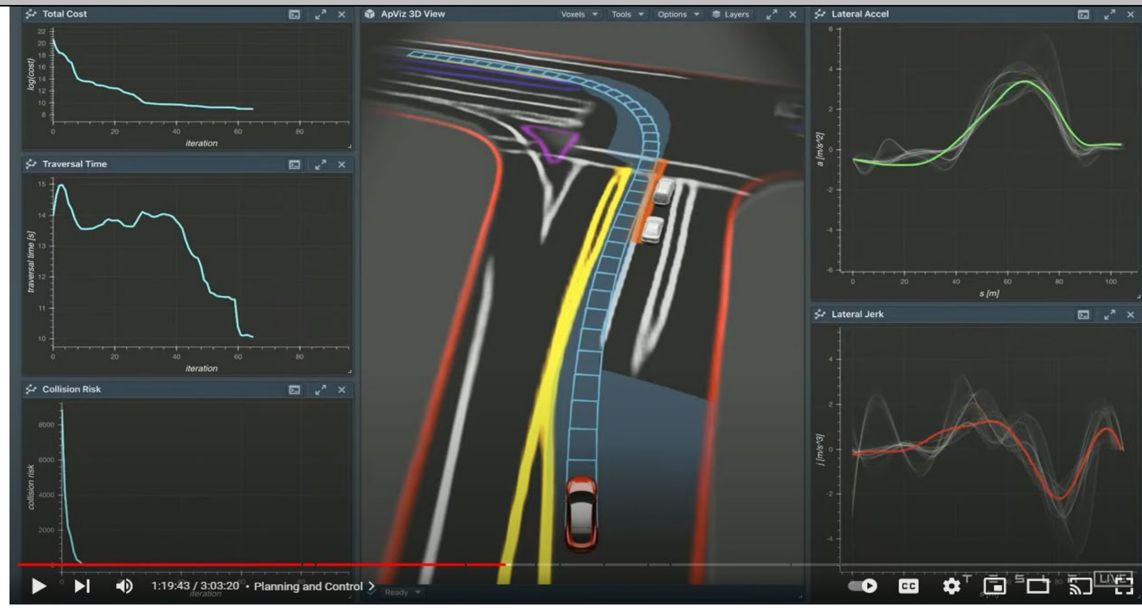
U.S. Patent No. 11,663,474	**Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]**
are applied to the first device.	stated that, "it is such a big deal that we have the fleet," and at 55:55, it is reiterated that, "Tesla is in such a unique and interesting position" because of the fleet. Finally, at/around 1:22:14, it is proclaimed that it "[i]s extremely difficult to catch up when Tesla has 100 times more miles per day than everyone else combined," which is a direct result of the fleet. *See also*, the evidence presented in support of [45a], [45f], and [45g] above.
56. The first device of claim 46, wherein the one or more inputs are further correlated with another one or more instruction sets for operating a third device, and wherein at least a portion of the one or more instruction sets for operating the second device or at least an information related to the one or more instruction sets for operating the second device is learned in a	The '474 Accused Products comprise the first device of claim 46, wherein the one or more inputs are further correlated with another one or more instruction sets for operating a third device, and wherein at least a portion of the one or more instruction sets for operating the second device or at least an information related to the one or more instruction sets for operating the second device is learned in a learning process that includes operating the second device at least partially by a user, and wherein at least a portion of the another one or more instruction sets for operating the third device or at least an information related to the another one or more instruction sets for operating the third device is learned in another learning process that includes operating the third device at least partially by another user. For example, a first, second, third, etc. circumstance representation derived from potentially multiple sensed inputs can correspond to/be correlated with a first, second, third, etc. instruction set. *See* the evidence presented in support of claim 45 above.

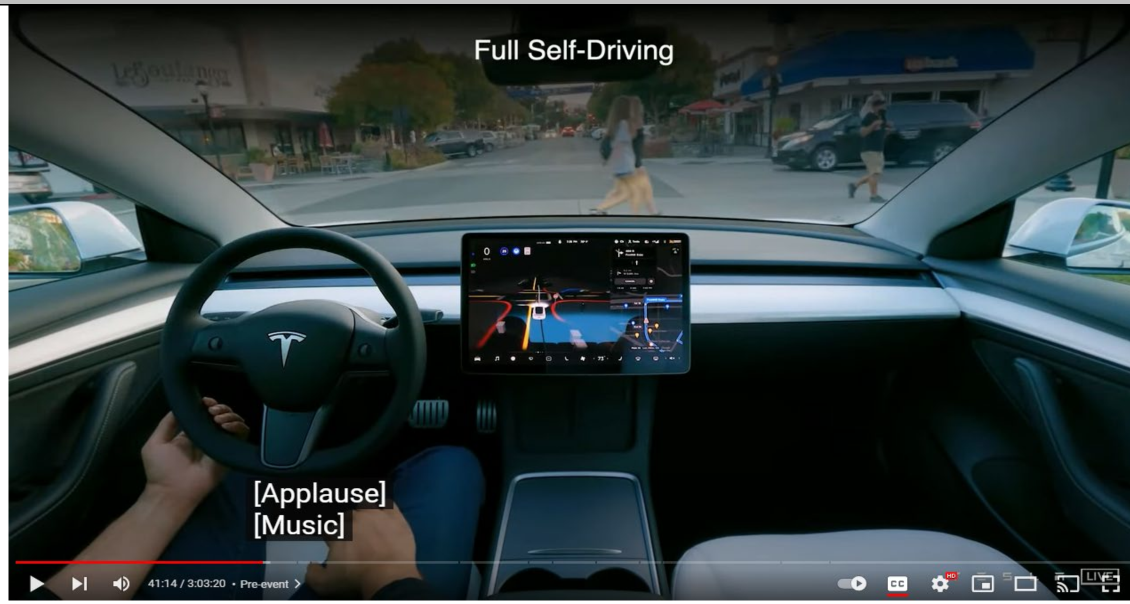
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
learning process that includes operating the second device at least partially by a user, and wherein at least a portion of the another one or more instruction sets for operating the third device or at least an information related to the another one or more instruction sets for operating the third device is learned in another learning process that includes operating the third device at least partially by another user.	
57. The first device of claim 46, wherein the one or more inputs are further correlated with another one or more instruction	The '474 Accused Products comprise the first device of claim 46, wherein the one or more inputs are further correlated with another one or more instruction sets for operating a third device. For example, an input of/a series of inputs to the 474 Accused Products can be correlated with a second, third, fourth, etc. instruction set(s) for operating a third, fourth, fifth, etc. device (e.g., Tesla vehicle).

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
sets for operating a third device.	*See* the evidence presented in claim 56 above. For example, the third device is a third Tesla vehicle and the one or more inputs are correlated with another one or more instruction sets for operating that third Tesla vehicle.
59. The first device of claim 46, wherein the one or more object representations of the generated circumstance representation include one or more information about one or more objects.	The '474 Accused Products comprise the first device of claim 46, wherein the one or more object representations of the generated circumstance representation include one or more information about one or more objects. For example, circumstance representations include information about objects, e.g., human, animal, vehicle, car, truck, speed, predicted path, etc. *See* the evidence presented in support of [45a] for more information on the relationship between circumstance representations and the object representations included therein.
60. The first device of claim 46, wherein the one or more object representations of the generated circumstance representation include one or more coordinates of one or more objects.	The '474 Accused Products comprise the first device of claim 46, wherein the one or more object representations of the generated circumstance representation include one or more coordinates of one or more objects. For example, the vehicle must know the distance from the detected object representations in the generated circumstance. The distance and location of the object representation in the circumstance representation is an example of a coordinate relative to the Telsa vehicle. *See* the evidence presented in support of claim 59 above. *See also*, Mr. Karpathy speaking during Tesla Autonomy Day 2019 "while you are driving a car what you're actually doing is you are annotating the data because you are steering the wheel. you're telling us how to traverse different environments so what we're looking at here is a some person in the fleet who took a left through an intersection and what we do here is ***we have the full video of all the cameras and we know that the path that this person took because of the GPS***, the inertial measurement unit, the wheel angle, the wheel ticks, so we put all that together and we understand the path that this person took through this environment and then of course we can use this for supervision [e.g., training a CNN

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	through supervised learning] for the network so we just source a lot of this from the fleet, we train a neural network on those trajectories, and then the neural network predicts paths just from that data so really what this is referred to typically is called imitation learning we're taking human trajectories from the real world I'm just trying to imitate how people drive in real worlds and we can also apply the same data engine crank to all of this and make this work [***emphasis*** added]**.**" Source: https://www.youtube.com/watch?v=-b041NXGPZ8 at 1:04:10. *See also*, para. [36] of Tesla's US 2020/0249685, which states, "At 303, data related to the elements of the time series are received. …In some embodiments, the related data is odometry data of the vehicle. … The related data may include vehicle operating parameters such as the speed, change in speed, acceleration, change in acceleration, steering, change in steering, braking, change in braking, etc." Further, Tesla's 10,956,755 explains that "…odometry data may include vehicle operation parameters such as applied acceleration, applied braking, applied steering, ***vehicle location, vehicle orientation, the change in vehicle location, the change in vehicle orientation***, etc. [***emphasis*** added]."
68. The first device of claim 46, wherein the knowledgebase is a neural network.	The '474 Accused Products comprise the first device of claim 46, wherein the knowledgebase is a neural network. For example, the 474 Accused Products implements a convolutional neural network. As Mr. Karpathy explained during Tesla Autonomy Day 2019 "while you are driving a car what you're actually doing is you are annotating the data because you are steering the wheel. you're telling us how to traverse different environments so what we're looking at here is a some person in the fleet who took a left through an intersection and what we do here is we have the full video of all the cameras and we know that the path that this person took because of the GPS, the inertial measurement unit, the wheel angle, the wheel ticks, so we put all that together and we understand the path that this person took through this environment and then of course we can use this for supervision [e.g., training a CNN through supervised learning] for the network so we just source a lot of this from the fleet, we train a neural network on those trajectories, and then the neural network predicts paths just from that data so really what ***this is referred to typically is called imitation learning we're taking human trajectories from the real world I'm just trying to imitate how people drive in real worlds and we can also apply the same data engine crank to all of this and make this work*** [***emphasis*** added]**.**" Source: https://www.youtube.com/watch?v=-b041NXGPZ8 at 1:04:10.

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
	See also, the evidence presented in support of [45a] above.
69. The first device of claim 46, wherein the knowledgebase includes an artificial intelligence system.	The '474 Accused Products comprise the first device of claim 46, wherein the knowledgebase includes an artificial intelligence system. For example, the supercomputer/simulator included in the 474 Accused Products is an artificial intelligence system (e.g., a convolutional neural network, or CNN). *See* the evidence presented in support of claim 68 above.
78. The first device of claim 46, wherein the one or more instruction sets for operating the second device are further for operating at least a third device.	The '474 Accused Products comprise the first device of claim 46, wherein For example, instruction sets derived from circumstance representations generated by the 474 Accused Products can be used to operate an entire fleet (e.g., tens/hundreds of thousands, even millions) of Tesla vehicles). The ability of vehicles in Tesla's fleet to learn from each other is widely acknowledged by Mr. Musk and Mr. Karpathy as one of Tesla's biggest competitive advantages. *See*, for example, Tesla Autonomy Day 2019 at/around 38:58, "I think a very powerful sustainable advantage for us is the fleet." At/around 44:40, it is stated that, "it is such a big deal that we have the fleet," and at 55:55, it is reiterated that, "Tesla is in such a unique and interesting position" because of the fleet. Finally, at/around 1:22:14, it is proclaimed that it "[i]s extremely difficult to catch up when Tesla has 100 times more miles per day than everyone else combined," which is a direct result of the fleet.
80. The first device of claim 46, wherein the generated circumstance representation represents the circumstance detected at least in part by the one or more sensors	The '474 Accused Products comprise the first device of claim 46, wherein the generated circumstance representation represents the circumstance detected at least in part by the one or more sensors during a time period. For example, Tesla vehicles are constantly sensing data (e.g., capturing images/video of the surrounding environment) and generating circumstance representations therefrom (e.g., via the supercomputer/simulator on the back-end of the 474 Accused Products). As an example, at/around 1:10:43 in Tesla's AI Day 2021 presentation, the speaker describes how Tesla depicts circumstance representations and the driving instructions (e.g., a trajectory through a turn) correlated therewith, along with how each is implemented as sensors (e.g., cameras) are capturing data over a series of pints in time as Tesla vehicles are being drive in real-time.

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
during a time period.	 *See also*, the remaining evidence in support of claim 45 (particularly [45a]) above.
81. The first device of claim 46, wherein the generated circumstance representation is a data structure that includes one or more data about the circumstance detected at least in part by the one or more sensors, and	The '474 Accused Products comprise the first device of claim 46, wherein the generated circumstance representation is a data structure that includes one or more data about the circumstance detected at least in part by the one or more sensors, and wherein the circumstance detected at least in part by the one or more sensors includes one or more objects detected at least in part by the one or more sensors: at a first time, or during a first time period. For example, at 58:10 in Tesla's AI Day 2021 presentation (https://www.youtube.com/watch?v=j0z4FweCy4M), the speaker describes taking all of the images captured/recorded by the cameras included in Tesla vehicles and simultaneously feeding them into a single neural net and directly outputting them into a vector space.

U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
wherein the circumstance detected at least in part by the one or more sensors includes one or more objects detected at least in part by the one or more sensors: at a first time, or during a first time period.	 *See also*, the additional evidence corresponding to convolutional neural network implemented by the 474 Accused Products, e.g., in claims 68 and 69 above.
[84pre]. A system comprising:	The '474 Accused Products are/include a system(s) comprising the limitations below. For example, *see* the evidence presented in support of claim [45pre] above.
[84a]. means for accessing a knowledgebase that includes one or more inputs for	The '474 Accused Products are/include a system(s) comprising means for accessing a knowledgebase that includes one or more inputs for inputting at least a portion of a circumstance representation, wherein the one or more inputs are correlated with one or more instruction sets for operating a first device. For example, *see* the evidence presented in support of [45a], [45c], and [45d] above.

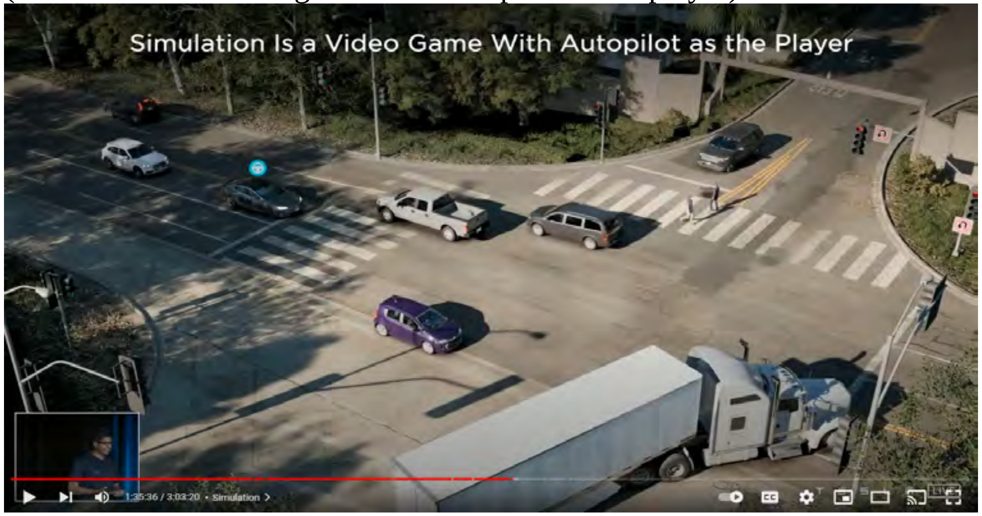
U.S. Patent No. 11,663,474	Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]
inputting at least a portion of a circumstance representation, wherein the one or more inputs are correlated with one or more instruction sets for operating a first device;	
[84b]. means for generating a circumstance representation, wherein the generated circumstance representation represents a circumstance detected at least in part by one or more sensors of a second device;	The '474 Accused Products are/include a system(s) further comprising means for generating a circumstance representation, wherein the generated circumstance representation represents a circumstance detected at least in part by one or more sensors of a second device. For example, *see* the evidence presented in support of [45c], [45d], and [45e] above.
[84c]. means for determining the one or more instruction sets for operating the first device at least by:	The '474 Accused Products are/include a system(s) further comprising means for determining the one or more instruction sets for operating the first device at least by: inputting at least a portion of the generated circumstance representation into the one or more inputs, and using a correlation between the one or more inputs and the one or more instruction sets for operating the first device. For example, *see* the evidence presented in support of [45c], [45d], and [45f] above.

U.S. Patent No. 11,663,474	**Exemplary Evidence of Plaintiff's Theory(ies) of Infringement[1] against the'474 Accused Products[2]**
inputting at least a portion of the generated circumstance representation into the one or more inputs, and using a correlation between the one or more inputs and the one or more instruction sets for operating the first device; and	
[84d]. means for executing, at least in response to the determining, the one or more instruction sets for operating the first device, wherein the second device autonomously performs one or more operations defined by the one or more instruction sets for operating the first device.	The '474 Accused Products are/include a system(s) further comprising means for executing, at least in response to the determining, the one or more instruction sets for operating the first device, wherein the second device autonomously performs one or more operations defined by the one or more instruction sets for operating the first device. For example, *see* the evidence presented in support of [45c], [45d], and [45g] above.

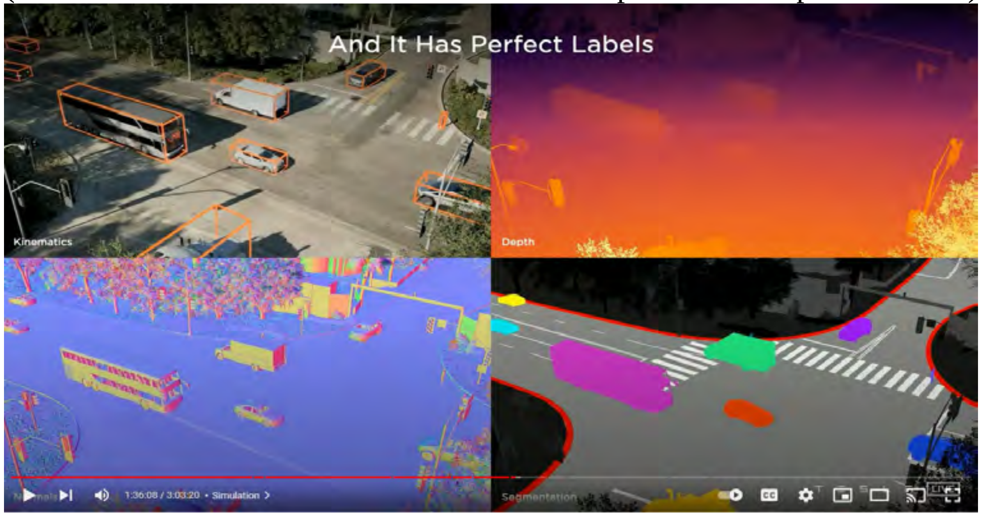
Exhibit L

U.S. Patent No. 10,607,134	
Claim 1	**Exemplary Infringement Evidence**[1]
[1pre] A system comprising:	The Dojo supercomputer infringes the '134 patent alone or together with Tesla vehicles operating Software Version 9.0 and beyond (this includes vehicles with enhanced autopilot and/or full self-driving (FSD)). The Dojo supercomputer alone or together with each of these vehicles meets the limitations of the claimed system. The discussion and evidence cited in claims [1a-e] are incorporated herein. - See Tesla Autonomy Day 2019 video https://www.youtube.com/watch?v=-b041NXGPZ8 at 53:32 (this is actually a screenshot of our own simulator. we use simulation extensively. we use it to develop and evaluate the software. we've also even used it for training quite successfully), 1:46:51 (this is taken from our actual simulation environment), 40:52 (we have quite a good a simulation too).  - See Tesla AI Day video https://www.youtube.com/watch?v=j0z4FweCy4M at 1:35:05

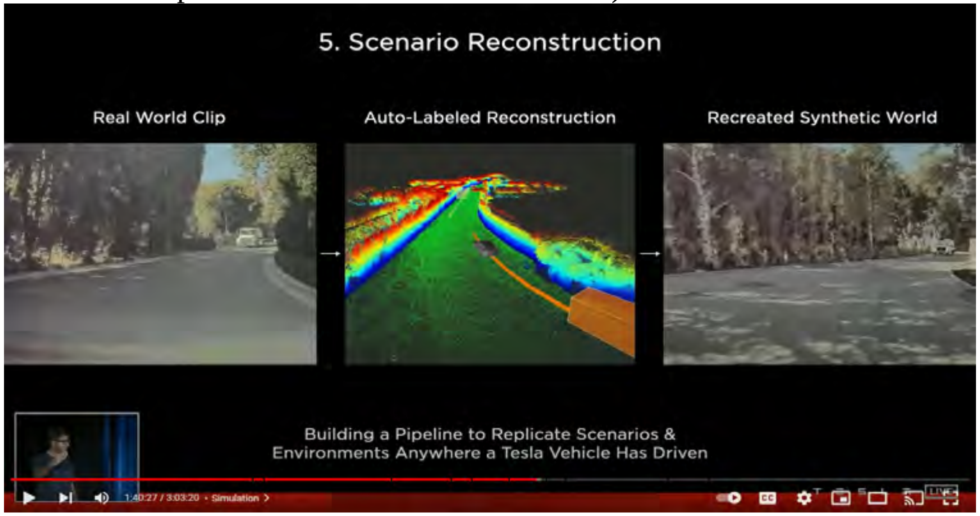
[1] These infringement contentions are prepared with publicly available information.

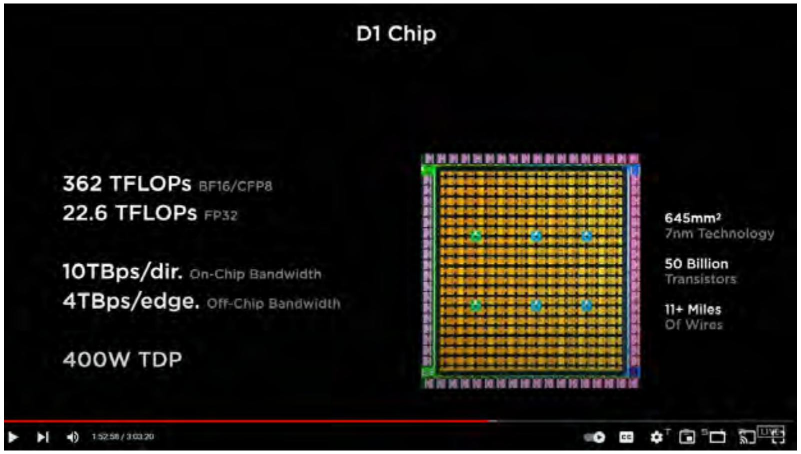
	(we also invest heavily in using simulation)  - See Tesla AI Day video https://www.youtube.com/watch?v=j0z4FweCy4M at 1:35:36 (Simulation is a video game with Autopilot as the player)  - See Tesla AI Day video https://www.youtube.com/watch?v=j0z4FweCy4M at 1:36:03

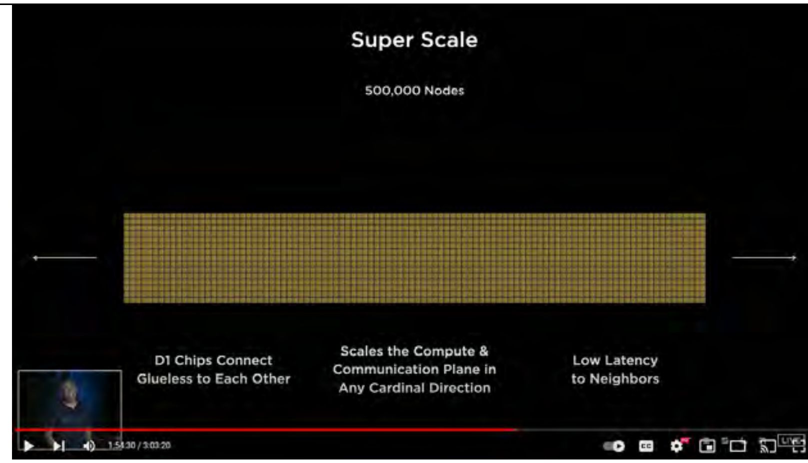
(since it's a simulation it starts from the vector space so it has perfect labels)



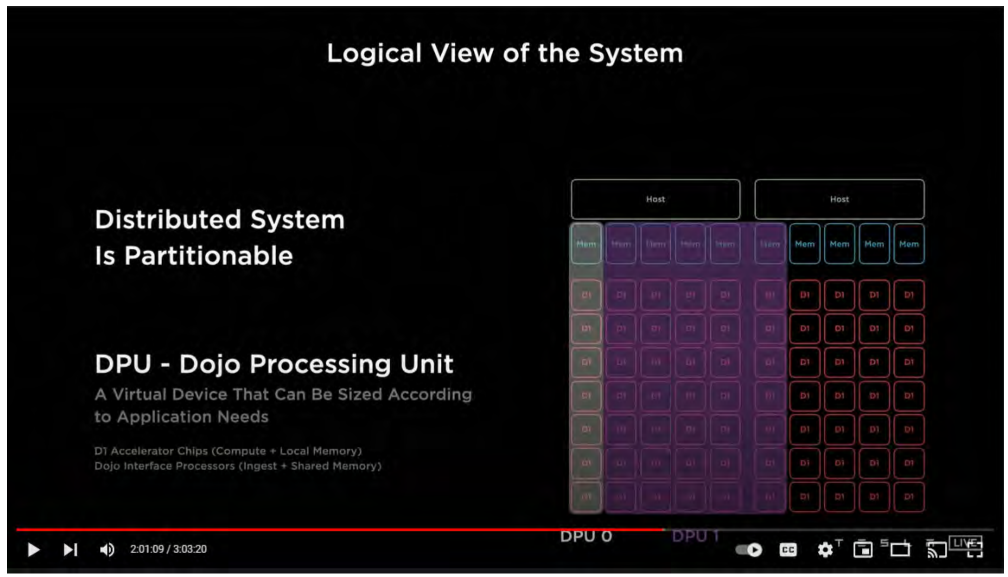
- See Tesla AI Day video https://www.youtube.com/watch?v=j0z4FweCy4M at 1:40:19 (we want to recreate any failures that happens to the autopilot in simulation so that we can hold autopilot to the same bar from then on)



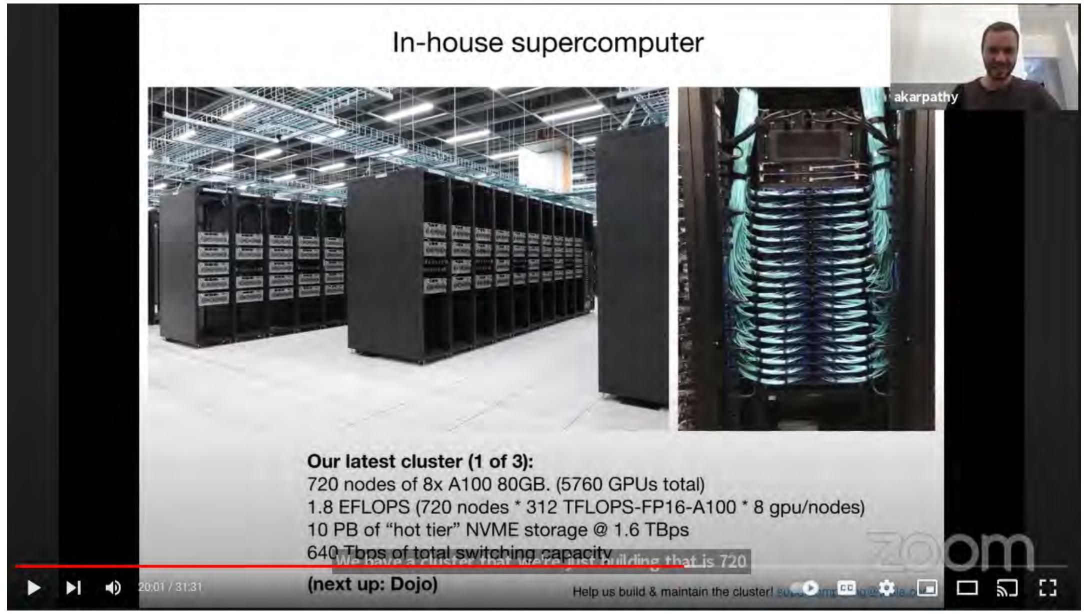
[1a] one or more processors; and	Tesla's autonomous vehicle simulation system includes one or more processors and one or more non-transitory machine readable media storing machine readable code that, when executed by the one or more processors, causes the one or more processors to perform the claimed features. For example, the Dojo supercomputer includes one or more processors including one or more Dojo D1 chip(s).  *See* Tesla AI Day video, available at https://www.youtube.com/watch?v=j0z4FweCy4M&t=3s

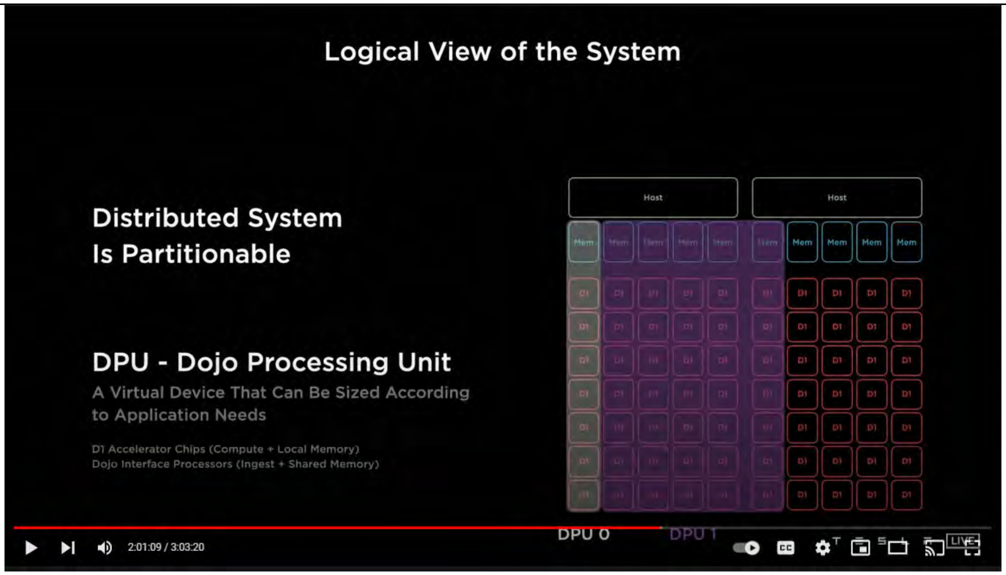
<table>
<tr>
<td></td>
<td>


Id.



Id.
</td>
</tr>
<tr>
<td>[1b] one or more non-transitory machine readable media storing machine readable code that, when executed by the one or more</td>
<td>Tesla's autonomous vehicle simulation system includes memory for storing digital pictures correlated with one or more instructions sets for operating a first object of a first application program. The discussion and evidence cited in claims [1c-f] is incorporated herein.</td>
</tr>
</table>

processors, causes the one or more processors to perform at least:	For example, Tesla's autonomous vehicle simulation system includes volatile and non-volatile storage, such as NVME or DRAM respectively, for storing executable machine readable code.  *See*, *e.g.*, [CVPR'21 WAD] Keynote - Andrej Karpathy, Tesla, available at https://www.youtube.com/watch?v=g6bOwQdCJrc.

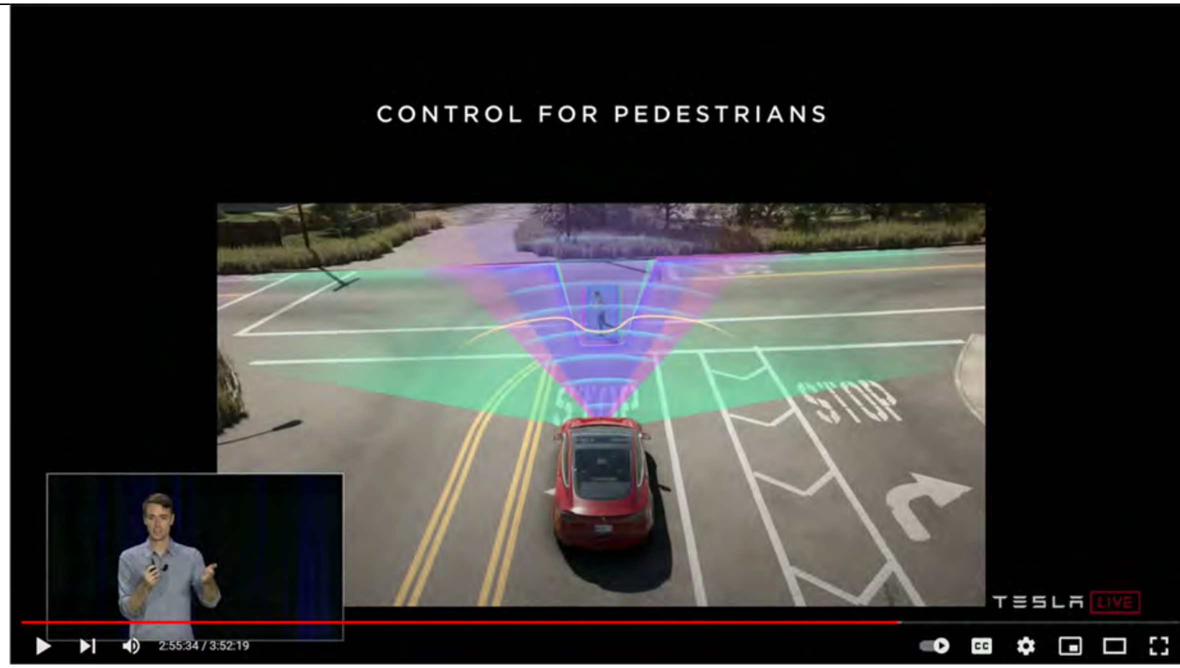


See Tesla AI Day video, available at https://www.youtube.com/watch?v=j0z4FweCy4M&t=3s

> *"We do have a major program at Tesla which we don't have enough time to talk about today called "Dojo". That's a super powerful training computer. The goal of Dojo will be to be able to take in vast amounts of data and train at a video level and do unsupervised massive training of vast amounts of video with the Dojo program – or Dojo computer."*

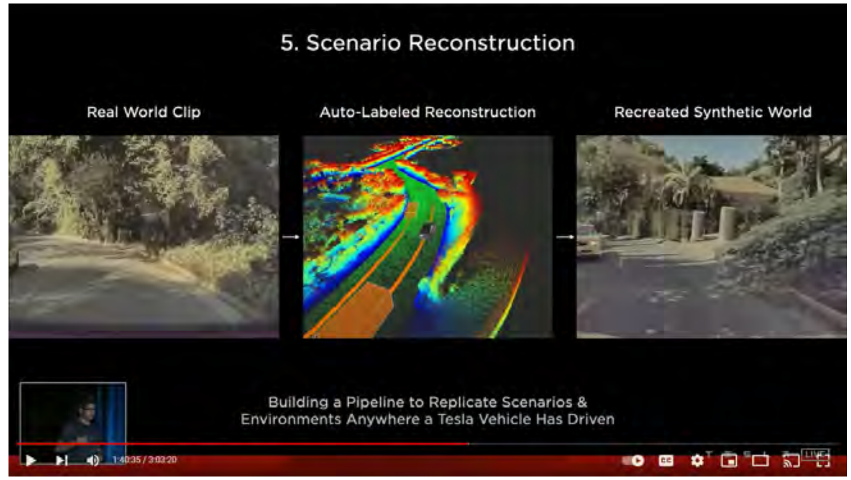
[Elon Musk hints at Tesla's not-so-secret Dojo AI-training supercomputer capacity - Electrek]

	Karpathy commented on the effort: > *"We have a neural net architecture network and we have a data set, a 1.5 petabytes data set that requires a huge amount of computing. So I wanted to give a plug to this insane supercomputer that we are building and using now. For us, computer vision is the bread and butter of what we do and what enables Autopilot. And for that to work really well, we need to master the data from the fleet, and train massive neural nets and experiment a lot. So we invested a lot into the compute. In this case, we have a cluster that we built with 720 nodes of 8x A100 of the 80GB version. So this is a massive supercomputer. I actually think that in terms of flops, it's roughly the number 5 supercomputer in the world."* https://www.inputmag.com/tech/tesla-showed-off-its-massive-supercomputer-for-self-driving-data-processing
[1c] accessing a first correlation including a first one or more object representations correlated with a first one or more instruction sets for operating a first avatar of an application, wherein the first one or more object representations represent one or more objects of the application;	Tesla's autonomous vehicle simulation system accesses a first correlation including a first one or more object representations correlated with a first one or more instruction sets for operating a first avatar of an application, and the first one or more object representations represent one or more objects of the application. For example, Tesla uses a supercomputer called DOJO to simulate real world driving environments using real world data and generated data. https://www.youtube.com/watch?v=g6bOwQdCJrc (14:30 to 15:30). Tesla's supercomputer is capable of behaving appropriately because the simulated car "avatar" is provided with driving instructions "first one or more instruction sets for operating a first avatar" that are correlated with object representations, e.g., simulated people, cars, traffic lights, traffic signs, etc. https://youtu.be/j0z4FweCy4M at 39 to 46 (showing simulated car in DOJO driving and reacting appropriately to cars, people, traffic signs, traffic lights, etc.); *see also* https://www.youtube.com/watch?v=6hkiTejoyms at 0:04 to 0:11 ("Today we are going to see how Tesla uses no less than a simulated game world to train their self-driving cars.").

<table>
<tr>
<td></td>
<td>

See https://www.youtube.com/watch?v=j0z4FweCy4M at 2:55:34 (discussing an automatic emergency breaking system). As shown in the clip, each simulated Tesla vehicle has sensors such as cameras, etc. that are used to detect a first object representations, e.g., a pedestrian in front of the vehicle.</td>
</tr>
<tr>
<td>[1d] generating or receiving a second one or more object representations, wherein the second one or more object representations represent one or more objects of the application;</td>
<td>Tesla's autonomous vehicle simulation system generates or receives a second one or more object representations, and the second one or more object representations represent one or more objects of the application. See citations and analysis for claim element 1b. The any object representation that is similar to but different from the first object representation can correspond to the claimed second "object representation," e.g., a second pedestrian different from the first different pedestrian, can represent an object (e.g., person) in the simulated application on the DOJO supercomputer.

For example, Tesla uses a supercomputer called DOJO to simulate real world driving environments for simulated Tesla Fleet vehicles using real world data and generated data.</td>
</tr>
</table>



See Tesla AI Day video, available at https://www.youtube.com/watch?v=j0z4FweCy4M&t=3s

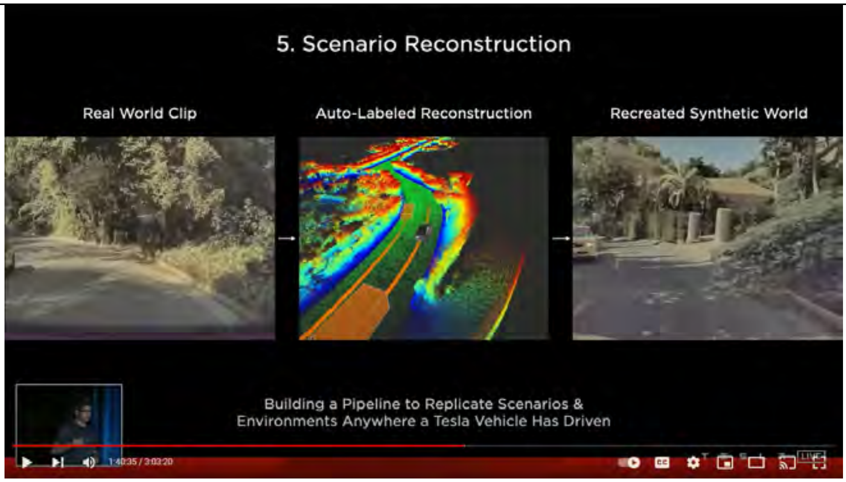
Id.

This real world and generated data includes video data, which includes object representations. Given that the video data is from the real world and generated simulations of real world

<table>
<tr>
<td></td>
<td>
environments the object representations represent objects of the application. See, for example, the simulated environment below showing a simulation of a person near the simulated car from the first application program (DOJO AI simulation).

See Tesla AI Day video, available at https://www.youtube.com/watch?v=j0z4FweCy4M&t=3s

As shown in the above-cited video, Tesla’s supercomputer is capable of behaving appropriately because the simulated car is provided with driving instructions that are correlated with objects, e.g., simulated people, cars, traffic lights, traffic signs, etc.
</td>
</tr>
</table>

	 *See* Tesla AI Day video, available at https://www.youtube.com/watch?v=j0z4FweCy4M&t=3s
[1e] determining the first one or more instruction sets for operating the first avatar of the application based on at least partial match between the second one or more object representations and the first one or more object representations; and	Tesla's autonomous vehicle simulation system determines the first one or more instruction sets for operating the first avatar of the application based on at least partial match between the second one or more object representations and the first one or more object representations. For example, the DOJO simulation determines an instruction set (e.g., steer to the left) for operating the first simulated vehicle (first avatar) of the simulation application based on at least partial match between the second one or more object representations (e.g., pedestrian jogging on the right hand side of the road) and the first one or more object representations (e.g., a pedestrian on the right hand side of the car). https://electrek.co/2021/08/19/watch-tesla-ai-day-livestream-important-news/ at 38:45 to 39:00
[1f] at least in response to the determining, causing the first avatar of the application or a second avatar of the application to perform one or more operations defined by the first one or more instruction sets for operating the first avatar of the application at	Tesla's autonomous vehicle simulation system causes, at least in response to the determining, the first avatar of the application or a second avatar of the application to perform one or more operations defined by the first one or more instruction sets for operating the first avatar of the application at least by executing the first one or more instruction sets for operating the first avatar of the application. For instance, the DOJO simulation, in response to determining the person of the right hand side of the simulated car, causes the first avatar (simulated car) of the application to perform one or more operations defined by the first one or more instruction sets for operating the first avatar of the application (i.e., steers the car to the left) at least by

least by executing the first one or more instruction sets for operating the first avatar of the application.	executing the first one or more instruction sets for operating the first avatar of the application. https://electrek.co/2021/08/19/watch-tesla-ai-day-livestream-important-news/ at 38:45 to 39:00.

Exhibit M

ABSTRACT

Aspects of the disclosure generally relate to computing enabled devices and/or systems, and may be generally directed to devices, systems, methods, and/or applications for learning a device's operation in various circumstances, storing this knowledge in a knowledgebase (i.e. neural network, graph, sequences, etc.), and enabling autonomous operation of the device.

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	
	Application Number	
Title of Invention	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION	

The application data sheet is part of the provisional or nonprovisional application for which it is being submitted. The following form contains the bibliographic data arranged in a format specified by the United States Patent and Trademark Office as outlined in 37 CFR 1.76.
This document may be completed electronically and submitted to the Office in electronic format using the Electronic Filing System (EFS) or the document may be printed and included in a paper filed application.

Secrecy Order 37 CFR 5.2:

☐ Portions or all of the application associated with this Application Data Sheet may fall under a Secrecy Order pursuant to 37 CFR 5.2 (Paper filers only. Applications that fall under Secrecy Order may not be filed electronically.)

Inventor Information:

Inventor 1 — Remove

Legal Name

Prefix	Given Name	Middle Name	Family Name	Suffix
	Jasmin		Cosic	

Residence Information (Select One) ● US Residency ○ Non US Residency ○ Active US Military Service

City	Miami	State/Province	FL	Country of Residence	US

Mailing Address of Inventor:

Address 1	108 Woodbury Street		
Address 2			
City	Pawtucket	State/Province	RI
Postal Code	02861	Country	US

All Inventors Must Be Listed - Additional Inventor Information blocks may be generated within this form by selecting the **Add** button. — Add

Correspondence Information:

Enter either Customer Number or complete the Correspondence Information section below. For further information see 37 CFR 1.33(a).

☐ **An Address is being provided for the correspondence Information of this application.**

Customer Number	116094
Email Address	cpapc29@hotmail.com

Add Email — Remove Email

Application Information:

Title of the Invention	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION		
Attorney Docket Number		Small Entity Status Claimed	☒
Application Type	Nonprovisional		
Subject Matter	Utility		
Total Number of Drawing Sheets (if any)	40	Suggested Figure for Publication (if any)	2

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	
	Application Number	
Title of Invention	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION	

Filing By Reference:

Only complete this section when filing an application by reference under 35 U.S.C. 111(c) and 37 CFR 1.57(a). Do not complete this section if application papers including a specification and any drawings are being filed. Any domestic benefit or foreign priority information must be provided in the appropriate section(s) below (i.e., "Domestic Benefit/National Stage Information" and "Foreign Priority Information").

For the purposes of a filing date under 37 CFR 1.53(b), the description and any drawings of the present application are replaced by this reference to the previously filed application, subject to conditions and requirements of 37 CFR 1.57(a).

Application number of the previously filed application	Filing date (YYYY-MM-DD)	Intellectual Property Authority or Country

Publication Information:

☐ Request Early Publication (Fee required at time of Request 37 CFR 1.219)

☒ **Request Not to Publish.** I hereby request that the attached application not be published under 35 U.S.C. 122(b) and certify that the invention disclosed in the attached application **has not and will not** be the subject of an application filed in another country, or under a multilateral international agreement, that requires publication at eighteen months after filing.

Representative Information:

Representative information should be provided for all practitioners having a power of attorney in the application. Providing this information in the Application Data Sheet does not constitute a power of attorney in the application (see 37 CFR 1.32).
Either enter Customer Number or complete the Representative Name section below. If both sections are completed the customer Number will be used for the Representative Information during processing.

Please Select One:	● Customer Number	US Patent Practitioner	○ Limited Recognition (37 CFR 11.9)
Customer Number			

Domestic Benefit/National Stage Information:

This section allows for the applicant to either claim benefit under 35 U.S.C. 119(e), 120, 121, 365(c), or 386(c) or indicate National Stage entry from a PCT application. Providing benefit claim information in the Application Data Sheet constitutes the specific reference required by 35 U.S.C. 119(e) or 120, and 37 CFR 1.78.
When referring to the current application, please leave the "Application Number" field blank.

Prior Application Status			Remove
Application Number	Continuity Type	Prior Application Number	Filing or 371(c) Date (YYYY-MM-DD)

Additional Domestic Benefit/National Stage Data may be generated within this form by selecting the **Add** button. Add

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	
	Application Number	
Title of Invention	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION	

Foreign Priority Information:

This section allows for the applicant to claim priority to a foreign application. Providing this information in the application data sheet constitutes the claim for priority as required by 35 U.S.C. 119(b) and 37 CFR 1.55. When priority is claimed to a foreign application that is eligible for retrieval under the priority document exchange program (PDX)[i] the information will be used by the Office to automatically attempt retrieval pursuant to 37 CFR 1.55(i)(1) and (2). Under the PDX program, applicant bears the ultimate responsibility for ensuring that a copy of the foreign application is received by the Office from the participating foreign intellectual property office, or a certified copy of the foreign priority application is filed, within the time period specified in 37 CFR 1.55(g)(1).

Application Number	Country[i]	Filing Date (YYYY-MM-DD)	Remove Access Code[i] (if applicable)

Additional Foreign Priority Data may be generated within this form by selecting the **Add** button.

Add

Statement under 37 CFR 1.55 or 1.78 for AIA (First Inventor to File) Transition Applications

☐ This application (1) claims priority to or the benefit of an application filed before March 16, 2013 and (2) also contains, or contained at any time, a claim to a claimed invention that has an effective filing date on or after March 16, 2013.
NOTE: By providing this statement under 37 CFR 1.55 or 1.78, this application, with a filing date on or after March 16, 2013, will be examined under the first inventor to file provisions of the AIA.

PTO/AIA/14 (11-15)
Approved for use through 09/30/2017. OMB 0651-0032
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it contains a valid OMB control number.

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	
	Application Number	

Title of Invention	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION

Authorization or Opt-Out of Authorization to Permit Access:

When this Application Data Sheet is properly signed and filed with the application, applicant has provided written authority to permit a participating foreign intellectual property (IP) office access to the instant application-as-filed (see paragraph A in subsection 1 below) and the European Patent Office (EPO) access to any search results from the instant application (see paragraph B in subsection 1 below).

Should applicant choose not to provide an authorization identified in subsection 1 below, applicant **must opt-out** of the authorization by checking the corresponding box A or B or both in subsection 2 below.

NOTE: This section of the Application Data Sheet is **ONLY** reviewed and processed with the **INITIAL** filing of an application. After the initial filing of an application, an Application Data Sheet cannot be used to provide or rescind authorization for access by a foreign IP office(s). Instead, Form PTO/SB/39 or PTO/SB/69 must be used as appropriate.

1. Authorization to Permit Access by a Foreign Intellectual Property Office(s)

A. Priority Document Exchange (PDX) - Unless box A in subsection 2 (opt-out of authorization) is checked, the undersigned hereby **grants the USPTO authority** to provide the European Patent Office (EPO), the Japan Patent Office (JPO), the Korean Intellectual Property Office (KIPO), the State Intellectual Property Office of the People's Republic of China (SIPO), the World Intellectual Property Organization (WIPO), and any other foreign intellectual property office participating with the USPTO in a bilateral or multilateral priority document exchange agreement in which a foreign application claiming priority to the instant patent application is filed, access to: (1) the instant patent application-as-filed and its related bibliographic data, (2) any foreign or domestic application to which priority or benefit is claimed by the instant application and its related bibliographic data, and (3) the date of filing of this Authorization. See 37 CFR 1.14(h)(1).

B. Search Results from U.S. Application to EPO - Unless box B in subsection 2 (opt-out of authorization) is checked, the undersigned hereby **grants the USPTO authority** to provide the EPO access to the bibliographic data and search results from the instant patent application when a European patent application claiming priority to the instant patent application is filed. See 37 CFR 1.14(h)(2).

The applicant is reminded that the EPO's Rule 141(1) EPC (European Patent Convention) requires applicants to submit a copy of search results from the instant application without delay in a European patent application that claims priority to the instant application.

2. Opt-Out of Authorizations to Permit Access by a Foreign Intellectual Property Office(s)

☒ A. Applicant **DOES NOT** authorize the USPTO to permit a participating foreign IP office access to the instant application-as-filed. If this box is checked, the USPTO will not be providing a participating foreign IP office with any documents and information identified in subsection 1A above.

☒ B. Applicant **DOES NOT** authorize the USPTO to transmit to the EPO any search results from the instant patent application. If this box is checked, the USPTO will not be providing the EPO with search results from the instant application.

NOTE: Once the application has published or is otherwise publicly available, the USPTO may provide access to the application in accordance with 37 CFR 1.14.

PTO/AIA/14 (11-15)
Approved for use through 04/30/2017. OMB 0651-0032
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	
	Application Number	
Title of Invention	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION	

Applicant Information:

Providing assignment information in this section does not substitute for compliance with any requirement of part 3 of Title 37 of CFR to have an assignment recorded by the Office.

Applicant 1 Remove

If the applicant is the inventor (or the remaining joint inventor or inventors under 37 CFR 1.45), this section should not be completed. The information to be provided in this section is the name and address of the legal representative who is the applicant under 37 CFR 1.43; or the name and address of the assignee, person to whom the inventor is under an obligation to assign the invention, or person who otherwise shows sufficient proprietary interest in the matter who is the applicant under 37 CFR 1.46. If the applicant is an applicant under 37 CFR 1.46 (assignee, person to whom the inventor is obligated to assign, or person who otherwise shows sufficient proprietary interest) together with one or more joint inventors, then the joint inventor or inventors who are also the applicant should be identified in this section.

Clear

Assignee	Legal Representative under 35 U.S.C. 117	Joint Inventor
Person to whom the inventor is obligated to assign.	Person who shows sufficient proprietary interest	

If applicant is the legal representative, indicate the authority to file the patent application, the inventor is:

Name of the Deceased or Legally Incapacitated Inventor:

If the Applicant is an Organization check here. ☐

Prefix	**Given Name**	Middle Name	**Family Name**	Suffix

Mailing Address Information For Applicant:

Address 1			
Address 2			
City		**State/Province**	
Country		Postal Code	
Phone Number		Fax Number	
Email Address			

Additional Applicant Data may be generated within this form by selecting the Add button. Add

Assignee Information including Non-Applicant Assignee Information:

Providing assignment information in this section does not substitute for compliance with any requirement of part 3 of Title 37 of CFR to have an assignment recorded by the Office.

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	
	Application Number	

Title of Invention	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION

Assignee 1

Complete this section if assignee information, including non-applicant assignee information, is desired to be included on the patent application publication. An assignee-applicant identified in the "Applicant Information" section will appear on the patent application publication as an applicant. For an assignee-applicant, complete this section only if identification as an assignee is also desired on the patent application publication.

Remove

If the Assignee or Non-Applicant Assignee is an Organization check here. ☐

Prefix	**Given Name**	Middle Name	**Family Name**	Suffix

Mailing Address Information For Assignee including Non-Applicant Assignee:

Address 1			
Address 2			
City		**State/Province**	
Country i		Postal Code	
Phone Number		Fax Number	
Email Address			

Additional Assignee or Non-Applicant Assignee Data may be generated within this form by selecting the Add button. Add

Signature:

Remove

NOTE: This Application Data Sheet must be signed in accordance with 37 CFR 1.33(b). **However, if this Application Data Sheet is submitted with the INITIAL filing of the application and either box A or B is not checked in subsection 2 of the "Authorization or Opt-Out of Authorization to Permit Access" section, then this form must also be signed in accordance with 37 CFR 1.14(c).**

This Application Data Sheet **must** be signed by a patent practitioner if one or more of the applicants is a **juristic entity** (e.g., corporation or association). If the applicant is two or more joint inventors, this form must be signed by a patent practitioner, **all** joint inventors who are the applicant, or one or more joint inventor-applicants who have been given power of attorney (e.g., see USPTO Form PTO/AIA/81) on behalf of **all** joint inventor-applicants.

See 37 CFR 1.4(d) for the manner of making signatures and certifications.

Signature	/Jasmin Cosic/			Date (YYYY-MM-DD)	2016-11-02
First Name	Jasmin	Last Name	Cosic	Registration Number	

Additional Signature may be generated within this form by selecting the Add button. Add

PTO/AIA/14 (11-15)
Approved for use through 04/30/2017. OMB 0651-0032
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	
	Application Number	

Title of Invention	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1 The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these records.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3 A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent CooperationTreaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

PTO/AIA/01 (06-12)
Approved for use through 01/31/2014. OMB 0651-0032
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

DECLARATION (37 CFR 1.63) FOR UTILITY OR DESIGN APPLICATION USING AN APPLICATION DATA SHEET (37 CFR 1.76)

Title of Invention	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION

As the below named inventor, I hereby declare that:

This declaration is directed to:

- [x] The attached application, or
- [ ] United States application or PCT international application number ____________________

filed on ____________________.

The above-identified application was made or authorized to be made by me.

I believe that I am the original inventor or an original joint inventor of a claimed invention in the application.

I hereby acknowledge that any willful false statement made in this declaration is punishable under 18 U.S.C. 1001 by fine or imprisonment of not more than five (5) years, or both.

WARNING:

Petitioner/applicant is cautioned to avoid submitting personal information in documents filed in a patent application that may contribute to identity theft. Personal information such as social security numbers, bank account numbers, or credit card numbers (other than a check or credit card authorization form PTO-2038 submitted for payment purposes) is never required by the USPTO to support a petition or an application. If this type of personal information is included in documents submitted to the USPTO, petitioners/applicants should consider redacting such personal information from the documents before submitting them to the USPTO. Petitioner/applicant is advised that the record of a patent application is available to the public after publication of the application (unless a non-publication request in compliance with 37 CFR 1.213(a) is made in the application) or issuance of a patent. Furthermore, the record from an abandoned application may also be available to the public if the application is referenced in a published application or an issued patent (see 37 CFR 1.14). Checks and credit card authorization forms PTO-2038 submitted for payment purposes are not retained in the application file and therefore are not publicly available.

LEGAL NAME OF INVENTOR

Inventor: Jasmin Cosic Date (Optional) : ____________________

Signature: ____________________

Note: An application data sheet (PTO/SB/14 or equivalent), including naming the entire inventive entity, must accompany this form or must have been previously filed. Use an additional PTO/AIA/01 form for each additional inventor.

ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION

FIELD

The disclosure generally relates to computing enabled devices and/or systems. The disclosure includes devices, apparatuses, systems, and related methods for providing advanced learning, anticipating, decision making, automation, and/or other functionalities.

COPYRIGHT NOTICE

BACKGROUND

Devices or systems commonly operate by receiving a user's operating directions in various circumstances. Instructions are then executed to effect the operation of a device or system based on user's operating directions. Hence, devices or systems rely on the user to direct their behaviors. Commonly employed device or system operating techniques lack a way to learn operation of a device or system and enable autonomous operation of a device or system.

SUMMARY

In some aspects, the disclosure relates to a system for learning and using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises a processor circuit configured to execute instruction sets for operating a device. The system may further include a memory unit configured to store data. The system may further include a sensor configured to detect objects. The system may further include an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first collection of object representations, the first collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to learn the first collection of object representations correlated with the first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: receive a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit,

wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations performed in response to the executing by the processor circuit.

In some embodiments, at least one of: the processor circuit, the memory unit, the sensor, or the artificial intelligence unit are part of, operating on, or coupled to the device. In further embodiments, the device includes one or more devices. In further embodiments, the device includes a construction machine, an assembly machine, an object handling machine, an object dispensing machine, a sorting machine, a restocking machine, an industrial machine, an agricultural machine, a harvesting machine, an appliance, a toy, a robot, a ground vehicle, an aerial vehicle, an aquatic vehicle, a computer, a smartphone, a control device, or a computing enabled device. In further embodiments, the processor circuit includes one or more processor circuits. In further embodiments, the processor circuit includes a logic circuit. The logic circuit may include a microcontroller. The one or more instruction sets may include one or more inputs into or one or more outputs from the logic circuit.

In certain embodiments, the processor circuit includes a logic circuit, the instruction sets for operating the device include inputs into the logic circuit, and executing instruction sets for operating the device includes performing logic operations on the inputs into the logic circuit and producing outputs for operating the device. The logic circuit includes a microcontroller.

In some embodiments, the processor circuit includes a logic circuit, the instruction sets for operating the device include outputs from the logic circuit for operating the device, and executing instruction sets for operating the device includes performing logic operations on inputs into the logic circuit and producing the outputs from the logic circuit for operating the device.

In certain embodiments, the memory unit includes one or more memory units. In further embodiments, the memory unit resides on a remote computing device or a remote computing system, the remote computing device or the remote computing system coupled to the processor circuit via a network or an interface. The remote computing device or the remote computing system may include a server, a cloud, a computing device, or a computing system accessible over the network or the interface.

In some embodiments, the sensor includes one or more sensors. In further embodiments, the sensor includes a camera, a microphone, a lidar, a radar, a sonar, or a detector. In further embodiments, the sensor is part of a remote device. In further embodiments, the sensor is configured to detect objects in the device's surrounding.

In certain embodiments, the artificial intelligence unit is coupled to the sensor. In further embodiments, the artificial intelligence unit is coupled to the memory unit. In further embodiments, the artificial intelligence unit is part of, operating on, or coupled to the processor circuit. In further embodiments, the system further comprises: an additional processor circuit, wherein the artificial intelligence unit is part of, operating on, or coupled to the additional processor circuit. In further embodiments, the artificial intelligence unit is part of, operating on, or coupled to a remote computing device or a remote computing system, the remote computing device or the remote computing system coupled to the processor circuit via a network or an interface. In further embodiments, the artificial intelligence unit includes a circuit, a computing apparatus, or a computing system attachable to the processor circuit. In further embodiments, the artificial intelligence unit includes a circuit, a computing apparatus, or a computing system attachable to the device. In further embodiments, the artificial intelligence unit is attachable to an application for operating the device, the application running on the processor circuit. In further embodiments, the artificial

intelligence unit includes a circuit, a computing apparatus, or a computing system built into the processor circuit. In further embodiments, the artificial intelligence unit includes a circuit, a computing apparatus, or a computing system built into the device. In further embodiments, the artificial intelligence unit is built into an application for operating the device, the application running on the processor circuit. In further embodiments, the artificial intelligence unit is provided as a feature of the processor circuit. In further embodiments, the artificial intelligence unit is provided as a feature of an application running on the processor circuit. In further embodiments, the artificial intelligence unit is provided as a feature of the device. In further embodiments, the artificial intelligence unit is further configured to: take control from, share control with, or release control to the processor circuit. In further embodiments, the artificial intelligence unit is further configured to: take control from, share control with, or release control to an application or an object of the application, the application running on the processor circuit.

In some embodiments, the first collection of object representations includes one or more representations of objects detected by the sensor at a time. In further embodiments, the new collection of object representations includes one or more representations of objects detected by the sensor at a time. In further embodiments, the first collection of object representations includes a stream of collections of object representations. In further embodiments, the new collection of object representations includes a stream of collections of object representations. In further embodiments, the first or the new collection of object representations includes or is associated with a time stamp, an order, or a time related information. In further embodiments, the first or the new collection of object representations includes one or more representations of objects in the device's surrounding. In further embodiments, the first or the new collection of object representations includes one or more representations of objects in a remote device's surrounding. In further embodiments, an object representation of the one or more object representations includes one or more object properties. In further embodiments, the first or the new collection of object representations includes one or more object properties. In further embodiments, the first collection of object representations includes a comparative collection of object representations whose at least one portion can be used for comparisons with at least one portion of collections of object representations subsequent to the first collection of object representations, the collections of object representations subsequent to the first collection of object representations comprising the new collection of object representations. In further embodiments, the first collection of object representations includes a comparative collection of object representations that can be used for comparisons with the new collection of object representations. In further embodiments, the new collection of object representations includes an anticipatory collection of object representations whose correlated one or more instruction sets can be used for anticipation of one or more instruction sets to be executed by the processor circuit.

In certain embodiments, the first one or more instruction sets for operating the device include one or more instruction sets that temporally correspond to the first collection of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first collection of object representations include one or more instruction sets executed at a time of generating the first collection of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first collection of object representations include one or more instruction sets executed prior to generating the first collection of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first collection of object representations include one or more instruction sets executed within a threshold period of time prior to generating the first collection of object representations. In further embodiments, the one or more instruction

sets that temporally correspond to the first collection of object representations include one or more instruction sets executed subsequent to generating the first collection of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first collection of object representations include one or more instruction sets executed within a threshold period of time subsequent to generating the first collection of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first collection of object representations include one or more instruction sets executed within a threshold period of time prior to generating the first collection of object representations or a threshold period of time subsequent to generating the first collection of object representations.

In some embodiments, the first one or more instruction sets for operating the device are executed by the processor circuit. In further embodiments, the first one or more instruction sets for operating the device are part of an application for operating the device, the application running on the processor circuit. In further embodiments, the first one or more instruction sets for operating the device include one or more inputs into or one or more outputs from the processor circuit. In further embodiments, the first one or more instruction sets for operating the device include values or states of one or more registers or elements of the processor circuit. In further embodiments, the first one or more instruction sets for operating the device include at least one of: a command, a keyword, a symbol, an instruction, an operator, a variable, a value, an object, a data structure, a function, a parameter, a state, a signal, an input, an output, a character, a digit, or a reference thereto. In further embodiments, the first one or more instruction sets for operating the device include a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, a structured query language (SQL) code, or a machine code. In further embodiments, the first one or more instruction sets for operating the device include one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the processor circuit includes a logic circuit. The first one or more instruction sets for operating the device may include one or more inputs into a logic circuit. The first one or more instruction sets for operating the device may include one or more outputs from a logic circuit.

In certain embodiments, the first one or more instruction sets for operating the device include one or more instruction sets for operating an application or an object of the application, the application running on the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes obtaining the first one or more instruction sets for operating the device from the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device as they are executed by the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from a register or an element of the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from an element that is part of, operating on, or coupled to the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from at least one of: the memory unit, the device, a virtual machine, a runtime engine, a hard drive, a storage device, a peripheral device, a network connected device, or a user. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from a plurality of

processor circuits, applications, memory units, devices, virtual machines, runtime engines, hard drives, storage devices, peripheral devices, network connected devices, or users.

In some embodiments, the processor circuit includes a logic circuit, and wherein the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from the logic circuit. In further embodiments, the logic circuit includes a microcontroller. In further embodiments, the receiving the first one or more instruction sets for operating the device from the logic circuit includes receiving the first one or more instruction sets for operating the device from an element of the logic circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device from the logic circuit includes receiving one or more inputs into the logic circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device from the logic circuit includes receiving one or more outputs from the logic circuit.

In certain embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from an application for operating the device, the application running on the processor circuit.

In some embodiments, the system further comprises: an application including instruction sets for operating the device, the application running on the processor circuit, wherein the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from the application.

In certain embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, a structured query language (SQL) code, or a machine code. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of an element that is part of, operating on, or coupled to the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a register of the processor circuit, the memory unit, a storage, or a repository where the first one or more instruction sets for operating the device are stored. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of the processor circuit, the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a processing element. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of the processor circuit or tracing, profiling, or instrumentation of a component of the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of an application or an object of the application, the application running on the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the

receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of one or more of code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a user input. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: a manual, an automatic, a dynamic, or a just in time (JIT) tracing, profiling, or instrumentation. In further embodiments, the receiving the first one or more instruction sets for operating the device includes utilizing at least one of: a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, a logging tool, or an independent tool for obtaining instruction sets. In further embodiments, the receiving the first one or more instruction sets for operating the device includes utilizing an assembly language. In further embodiments, the receiving the first one or more instruction sets for operating the device includes utilizing a branch or a jump. In further embodiments, the receiving the first one or more instruction sets for operating the device includes a branch tracing or a simulation tracing.

In further embodiments, the system further comprises: an interface configured to receive instruction sets, wherein the first one or more instruction sets for operating the device are received by the interface. The interface may include an acquisition interface.

In some embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the device include a knowledge of how the device operated in a circumstance. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the device are included in a neuron, a node, a vertex, or an element of a knowledgebase. In further embodiments, the knowledgebase includes a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. In further embodiments, some of the neurons, nodes, vertices, or elements are interconnected. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the device are structured into a knowledge cell. In further embodiments, the knowledge cell is included in a neuron, a node, a vertex, or an element of a knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device includes correlating the first collection of object representations with the first one or more instruction sets for operating the device. In further embodiments, the correlating the first collection of object representations with the first one or more instruction sets for operating the device includes generating a knowledge cell, the knowledge cell comprising the first collection of object representations correlated with the first one or more instruction sets for operating the device. In further embodiments, the correlating the first collection of object representations with the first one or more instruction sets for operating the device includes structuring a knowledge of how the device operated in a circumstance. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device includes learning a user's knowledge, style, or methodology of operating the device in a circumstance.

In certain embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device includes storing the first collection of object representations

correlated with the first one or more instruction sets for operating the device into the memory unit, the first collection of object representations correlated with the first one or more instruction sets for operating the device being part of a plurality of collections of object representations correlated with one or more instruction sets for operating the device stored in the memory unit. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the device include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the device are organized into a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, each of the plurality of collections of object representations correlated with one or more instruction sets for operating the device is included in a neuron, a node, a vertex, or an element of a knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the device include a user's knowledge, style, or methodology of operating the device in circumstances. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the device are stored on a remote computing device or a remote computing system. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the device include an artificial intelligence system for knowledge structuring, storing, or representation. The artificial intelligence system for knowledge structuring, storing, or representation may include at least one of: a deep learning system, a supervised learning system, an unsupervised learning system, a neural network, a search-based system, an optimization-based system, a logic-based system, a fuzzy logic-based system, a tree-based system, a graph-based system, a hierarchical system, a symbolic system, a sub-symbolic system, an evolutionary system, a genetic system, a multi-agent system, a deterministic system, a probabilistic system, or a statistical system.

In some embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes comparing at least one portion of the new collection of object representations with at least one portion of the first collection of object representations. In further embodiments, the at least one portion of the new collection of object representations include at least one object representation or at least one object property of the new collection of object representations. In further embodiments, the at least one portion of the first collection of object representations include at least one object representation or at least one object property of the first collection of object representations. In further embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes comparing at least one object representation from the new collection of object representations with at least one object representation from the first collection of object representations. In further embodiments, the comparing at least one object representation from the new collection of object representations with at least one object representation from the first

collection of object representations includes comparing at least one object property of the at least one object representation from the new collection of object representations with at least one object property of the at least one object representation from the first collection of object representations.

In certain embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes determining that there is at least a partial match between the new collection of object representations and the first collection of object representations. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that there is at least a partial match between one or more portions of the new collection of object representations and one or more portions of the first collection of object representations. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that a similarity between at least one portion of the new collection of object representations and at least one portion of the first collection of object representations exceeds a similarity threshold. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining a substantial similarity between at least one portion of the new collection of object representations and at least one portion of the first collection of object representations. The substantial similarity may be achieved when a similarity between the at least one portion of the new collection of object representations and the at least one portion of the first collection of object representations exceeds a similarity threshold. The substantial similarity may be achieved when a number or a percentage of matching or partially matching portions of the new collection of object representations and portions of the first collection of object representations exceeds a threshold number or threshold percentage. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that a number or a percentage of matching or partially matching object representations from the new collection of object representations and from the first collection of object representations exceeds a threshold number or threshold percentage. The matching or partially matching object representations from the new collection of object representations and from the first collection of object representations may be determined factoring in at least one of: a type of an object representation, an importance of an object representation, a threshold for a similarity in an object representation, or a threshold for a difference in an object representation. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that a number or a percentage of matching or partially matching object properties from the new collection of object representations and from the first collection of object representations exceeds a threshold number or threshold percentage. The matching or partially matching object properties from the new collection of object representations and from the first collection of object representations may be determined factoring in at least one of: an association of an object property with an object representation, a category of an object property, an importance of an object property, a threshold for a similarity in an object property, or a threshold for a difference in an object property. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that there

is at least a partial match between at least one object representation from the new collection of object representations and at least one object representation from the first collection of object representations. The determining that there is at least a partial match between at least one object representation from the new collection of object representations and at least one object representation from the first collection of object representations includes determining that there is at least a partial match between at least one object property of the at least one object representation from the new collection of object representations and at least one object property of the at least one object representation from the first collection of object representations.

In certain embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations instead of or prior to an instruction set that would have been executed next. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first collection of object representations into a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting the processor circuit to the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting the processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes transmitting, to the processor circuit for execution, the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes issuing an interrupt to the processor circuit and executing the first one or more instruction sets for operating the device correlated with the first collection of object representations following the interrupt. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying an element that is part of, operating on, or coupled to the processor circuit.

In some embodiments, the processor circuit includes a logic circuit, and wherein the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the logic circuit

includes a microcontroller. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying an element of the logic circuit. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first collection of object representations into an element of the logic circuit. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting the logic circuit to the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes replacing inputs into the logic circuit with the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes replacing outputs from the logic circuit with the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In certain embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes causing an application for operating the device to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations, the application running on the processor circuit.

In some embodiments, the system further comprises: an application including instruction sets for operating the device, the application running on the processor circuit, wherein the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying the application.

In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting an application to the first one or more instruction sets for operating the device correlated with the first collection of object representations, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting an application to one or more alternate instruction sets, the application running on the processor circuit, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying at least one of: the memory unit, a register of the processor circuit, a

storage, or a repository where instruction sets are stored or used. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets for operating an application or an object of the application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying at least one of: an element of the processor circuit, an element of the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing an assembly language. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes adding or inserting additional code into a code of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or

more instruction sets for operating the device correlated with the first collection of object representations includes at least one of: modifying, removing, rewriting, or overwriting a code of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of an application, the application running on the processor circuit. The branching or redirecting the code may include inserting at least one of: a branch, a jump, or a means for redirecting an execution. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes implementing a user's knowledge, style, or methodology of operating the device in a circumstance.

In certain embodiments, the system further comprises: an interface configured to cause execution of instruction sets, wherein the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations is caused by the interface. The interface may include a modification interface.

In some embodiments, the one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations include one or more operations with or by a computing enabled device. In further embodiments, the performing the one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations includes implementing a user's knowledge, style, or methodology of operating the device in a circumstance.

In certain embodiments, the system further comprising: an application running on the processor circuit.

In some embodiments, the instruction sets for operating the device are part of an application for operating the device, the application running on the processor circuit.

In certain embodiments, the system of further comprises: an application for operating the device, the application running on the processor circuit. The application for operating the device may include the instruction sets for operating the device.

In some embodiments, the artificial intelligence unit is further configured to: receive at least one extra information. In further embodiments, the at least one extra information include one or more of: a time information, a location information, a computed information, or a contextual information. In further embodiments, the at least one extra information include one or more of: an information on an object, an information on an object representation, an information on a collection of object representations, an information on a device's circumstance, an information on an instruction set, an information on an application, an information on the processor circuit, an information on the device, or an information on an user. In further embodiments, the artificial intelligence unit is further configured to: learn the first collection of object representations correlated with the at least one extra information. The learning the first collection of object representations correlated with at least one extra information may include correlating the first collection of object representations with the at least one extra information. The learning the first collection of object representations correlated with at least one extra information may include storing the first collection of object representations correlated with the at least one extra information into the memory unit. In further embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first

collection of object representations includes anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations. The anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations may include comparing an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations. The anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations may include determining that a similarity between an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations exceeds a similarity threshold.

In certain embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: cause the user interface to present a user with an option to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In some embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: receive, via the user interface, a user's selection to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In certain embodiments, the artificial intelligence unit is further configured to: rate the executed first one or more instruction sets for operating the device correlated with the first collection of object representations. The rating the executed first one or more instruction sets for operating the device correlated with the first collection of object representations may include causing a user interface to display the executed first one or more instruction sets for operating the device correlated with the first collection of object representations along with one or more rating values as options to be selected by a user. The rating the executed first one or more instruction sets for operating the device correlated with the first collection of object representations may include rating the executed first one or more instruction sets for operating the device correlated with the first collection of object representations without a user input.

In some embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: cause the user interface to present a user with an option to cancel the execution of the executed first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the canceling the execution of the executed first one or more instruction sets for operating the device correlated with the first collection of object representations includes restoring the processor circuit or the device to a prior state. The restoring the processor circuit or the device to a prior state may include saving the state of the processor circuit or the device prior to executing the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In certain embodiments, the system further comprises: an input device configured to receive a user's operating directions, the user's operating directions for instructing the processor circuit on how to operate the device.

In some embodiments, the autonomous device operating includes a partially or a fully autonomous device operating. The partially autonomous device operating may include executing the first one or more instruction sets for operating the device correlated with the first collection of object representations responsive to a user confirmation. In further embodiments, the fully autonomous device operating may include executing the first one or more instruction sets for operating the device correlated with the first collection of object representations without a user confirmation.

In certain embodiments, the artificial intelligence unit is further configured to: receive a second collection of object representations, the second collection of object representations including one or more representations of objects detected by the sensor; receive a second one or more instruction sets for operating the device; and learn the second collection of object representations correlated with the second one or more instruction sets for operating the device. In further embodiments, the second collection of object representations includes one or more representations of objects detected by the sensor at a time. In further embodiments, the second collection of object representations includes a stream of collections of object representations. In further embodiments, the second collection of object representations includes or is associated with a time stamp, an order, or a time related information. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device and the learning the second collection of object representations correlated with the second one or more instruction sets for operating the device include creating a connection between the first collection of object representations correlated with the first one or more instruction sets for operating the device and the second collection of object representations correlated with the second one or more instruction sets for operating the device. The connection may include or be associated with at least one of: an occurrence count, a weight, a parameter, or a data. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device and the learning the second collection of object representations correlated with the second one or more instruction sets for operating the device include updating a connection between the first collection of object representations correlated with the first one or more instruction sets for operating the device and the second collection of object representations correlated with the second one or more instruction sets for operating the device. The updating the connection between the first collection of object representations correlated with the first one or more instruction sets for operating the device and the second collection of object representations correlated with the second one or more instruction sets for operating the device may include updating at least one of: an occurrence count, a weight, a parameter, or a data included in or associated with the connection. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device includes storing the first collection of object representations correlated with the first one or more instruction sets for operating the device into a first node of a knowledgebase, and wherein the learning the second collection of object representations correlated with the second one or more instruction sets for operating the device includes storing the second collection of object representations correlated with the second one or more instruction sets for operating the device into a second node of the knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. The knowledgebase may be stored in the memory unit. The learning the first collection of object representations correlated with the first one or more instruction sets for operating the device and the learning the second collection of object representations correlated with the second one or more instruction sets for operating the device include creating a connection

between the first node and the second node. The learning the first collection of object representations correlated with the first one or more instruction sets for operating the device and the learning the second collection of object representations correlated with the second one or more instruction sets for operating the device include updating a connection between the first node and the second node. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the device is stored into a first node of a neural network and the second collection of object representations correlated with the second one or more instruction sets for operating the device is stored into a second node of the neural network. The first node and the second node may be connected by a connection. The first node may be part of a first layer of the neural network and the second node may be part of a second layer of the neural network. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the device is stored into a first node of a graph and the second collection of object representations correlated with the second one or more instruction sets for operating the device is stored into a second node of the graph. The first node and the second node may be connected by a connection. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the device is stored into a first node of a sequence and the second collection of object representations correlated with the second one or more instruction sets for operating the device is stored into a second node of the sequence.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first collection of object representations, the first collection of object representations including one or more representations of objects detected by a sensor. The operations may further include: receiving a first one or more instruction sets for operating a device. The operations may further include: learning the first collection of object representations correlated with the first one or more instruction sets for operating the device. The operations may further include: receiving a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The operations may further include: anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further include: causing an execution of the first one or more instruction sets for operating the device correlated with the first collection of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations by the device performed in response to the execution.

In certain embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from the one or more processor circuits or from another one or more processor circuits. In further embodiments, the execution of the first one or more instruction sets for operating the device correlated with the first collection of object representations is performed by the one or more processor circuits or by another one or more processor circuits.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a processor circuit, the first collection of object representations including one or more representations of objects detected by a sensor. The method may further include: (b) receiving a first one or more instruction sets for operating a device by the processor circuit. The method may further include: (c) learning the first collection of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the processor circuit. The method may further include: (d) receiving a new collection of object representations by the processor circuit, the new collection of object representations including one or more representations of objects detected by the sensor. The method may further include: (e) anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (e) performed by the processor circuit. The method may further include: (f) executing the first one or more instruction sets for operating the device correlated with the first collection of object representations, the executing of (f) performed in response to the anticipating of (e). The method may further include: (g) performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations by the device performed in response to the executing of (f).

In certain embodiments, the receiving of (b) includes receiving the first one or more instruction sets for operating the device from the processor circuit or from another processor circuit. In further embodiments, the executing of (f) is performed by the processor circuit or by another processor circuit.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable as well as the following embodiments.

In some embodiments, the first one or more instruction sets for operating the device are executed by a processor circuit. In further embodiments, the first one or more instruction sets for operating the device are part of an application for operating the device. In further embodiments, the first one or more instruction sets for operating the device include one or more inputs into or one or more outputs from a processor circuit. In further embodiments, the first one or more instruction sets for operating the device include values or states of one or more registers or elements of a processor circuit. In further embodiments, the first one or more instruction sets for operating the device include one or more inputs into a logic circuit. In further embodiments, the first one or more instruction sets for operating the device include one or more outputs from a logic circuit. In further embodiments, the first one or more instruction sets for operating the device include one or more instruction sets for operating an application or an object of the application.

In certain embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device as they are executed by a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first

one or more instruction sets for operating the device from a register or an element of a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from an element that is part of, operating on, or coupled to a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from at least one of: the memory unit, the device, a virtual machine, a runtime engine, a hard drive, a storage device, a peripheral device, a network connected device, or a user. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from a plurality of processor circuits, applications, memory units, devices, virtual machines, runtime engines, hard drives, storage devices, peripheral devices, network connected devices, or users.

In some embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from a logic circuit. In further embodiments, the logic circuit includes a microcontroller. In further embodiments, the receiving the first one or more instruction sets for operating the device from the logic circuit includes receiving the first one or more instruction sets for operating the device from an element of the logic circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device from the logic circuit includes receiving one or more inputs into the logic circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device from the logic circuit includes receiving one or more outputs from the logic circuit.

In certain embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from an application for operating the device. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from an application, the application including instruction sets for operating the device. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of an element that is part of, operating on, or coupled to a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a register of a processor circuit, the memory unit, a storage, or a repository where the first one or more instruction sets for operating the device are stored. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a processor circuit, the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a processing element. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a processor circuit or tracing, profiling, or instrumentation of a component of a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of an application or an object of the application. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device by an interface. The interface may include an acquisition interface.

In some embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device includes storing the first collection of object representations

correlated with the first one or more instruction sets for operating the device into a memory unit, the first collection of object representations correlated with the first one or more instruction sets for operating the device being part of a plurality of collections of object representations correlated with one or more instruction sets for operating the device stored in the memory unit.

In certain embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes executing the first one or more instruction sets for operating the device correlated with the first collection of object representations instead of or prior to an instruction set that would have been executed next. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying a register or an element of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first collection of object representations into a register or an element of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting a processor circuit to the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting a processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes transmitting, to a processor circuit for execution, the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes issuing an interrupt to a processor circuit and executing the first one or more instruction sets for operating the device correlated with the first collection of object representations following the interrupt. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying an element that is part of, operating on, or coupled to a processor circuit.

In some embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes executing, by a logic circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the logic circuit includes a microcontroller. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying an element of the logic circuit. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first collection of object representations into an element of the logic circuit. In further embodiments, the executing, by

the logic circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting the logic circuit to the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations includes replacing inputs into the logic circuit with the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations includes replacing outputs from the logic circuit with the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In certain embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes executing, by an application for operating the device, the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying an application, the application including instruction sets for operating the device. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting an application to the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting an application to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying at least one of: the memory unit, a register of a processor circuit, a storage, or a repository where instruction sets are stored or used. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets for operating an application or an object of the application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying at least one of: an element of a processor circuit, an element of the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more code segments, lines of

code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing an assembly language. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes adding or inserting additional code into a code of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes at least one of: modifying, removing, rewriting, or overwriting a code of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of an application. The branching or redirecting the code may include inserting at least one of: a branch, a jump, or a means for redirecting an execution. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes implementing a user's knowledge, style, or methodology of operating the device in a circumstance. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes executing the first one or more instruction sets for operating the device correlated with the first collection of object representations via an interface. The interface may include a modification interface.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving at least one extra information. In further embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: learning the first collection of object representations correlated with the at least one extra information.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: presenting, via a user interface, a user with an option to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving, via a user interface, a user's selection to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: rating the executed first one or more instruction sets for operating the device correlated with the first collection of object representations.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: presenting, via a user interface, a user with an option to cancel the execution of the executed first one or more instruction sets for operating the device correlated with the first collection of object representations.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving, via an input device, a user's operating directions, the user's operating directions for instructing a processor circuit on how to operate the device.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving a second collection of object representations, the second collection of object representations including one or more representations of objects detected by the sensor; receiving a second one or more instruction sets for operating the device; and learning the second collection of object representations correlated with the second one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving, by a first processor circuit of the one or more processor circuits, a first collection of object representations, the first collection of object representations including one or more representations of objects detected by a sensor. The operations may further comprise: receiving, by the first processor circuit of the one or more processor circuits, a first one or more instruction sets for operating a device. The operations may further comprise: learning, by the first processor circuit of the one or more processor circuits, the first collection of object representations correlated with the first one or more instruction sets for operating the device. The operations may further comprise: receiving, by the first processor circuit of the one or more processor circuits, a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The operations may further comprise: anticipating, by the first processor circuit of the one or more processor circuits, the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further comprise: causing, by the first processor circuit of the one or more processor circuits, an execution, by a second processor circuit of the one or more processor circuits, of the first one or more instruction sets for operating the device correlated with the first collection of object representations, the causing performed in response to the anticipating, by the first processor circuit of the one or more processor circuits, the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least

a partial match between the new collection of object representations and the first collection of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations by the device performed in response to the execution.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a first processor circuit, the first collection of object representations including one or more representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more instruction sets for operating a device by the first processor circuit. The method may further comprise: (c) learning the first collection of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the first processor circuit. The method may further comprise: (d) receiving a new collection of object representations by the first processor circuit, the new collection of object representations including one or more representations of objects detected by the sensor. The method may further comprise: (e) anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (e) performed by the first processor circuit. The method may further comprise: (f) executing, by a second processor circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations, the executing of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations by the device performed in response to the executing of (f).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for learning a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first collection of object representations, the first collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: learn the first collection of object representations correlated with the first one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first collection of object representations, the first collection of object representations including one or more representations of objects detected by a sensor. The operations may further comprise: receiving a first one or more instruction sets for

operating a device. The operations may further comprise: learning the first collection of object representations correlated with the first one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a processor circuit, the first collection of object representations including one or more representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more instruction sets for operating a device by the processor circuit. The method may further comprise: (c) learning the first collection of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the processor circuit.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: access the memory unit that comprises a plurality of collections of object representations correlated with one or more instruction sets for operating the device, the plurality of collections of object representations correlated with one or more instruction sets for operating the device including a first collection of object representations correlated with a first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: receive a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations performed in response to the executing by the processor circuit.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: accessing a memory unit that comprises a plurality of collections of object representations correlated with one or more instruction sets for operating a device, the plurality of collections of object representations correlated with one or more instruction sets for operating the device including a first collection of object representations correlated with a first one or more instruction sets for operating the device. The operations may further comprise: receiving a new collection of object representations, the new collection of object representations including one or more representations of objects detected by a sensor. The

operations may further comprise: anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further comprise: causing an execution of the first one or more instruction sets for operating the device correlated with the first collection of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations by the device performed in response to the execution.

In some aspects, the disclosure relates to a method comprising: (a) accessing a memory unit that comprises a plurality of collections of object representations correlated with one or more instruction sets for operating a device, the plurality of collections of object representations correlated with one or more instruction sets for operating the device including a first collection of object representations correlated with a first one or more instruction sets for operating the device, the accessing of (a) performed by a processor circuit. The method may further comprise: (b) receiving a new collection of object representations by the processor circuit, the new collection of object representations including one or more representations of objects detected by a sensor. The method may further comprise: (c) anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (c) performed by the processor circuit. The method may further comprise: (d) executing the first one or more instruction sets for operating the device correlated with the first collection of object representations, the executing of (d) performed in response to the anticipating of (c). The method may further comprise: (e) performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations by the device performed in response to the executing of (d).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for learning and using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first stream of collections of object representations, the first stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: learn the first stream of collections of object representations correlated with the first one or more instruction sets for operating the

device. The artificial intelligence unit may be further configured to: receive a new stream of collections of object representations, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations performed in response to the executing by the processor circuit.

In certain embodiments, each collection of object representations includes one or more representations of objects detected by the sensor at a time. In further embodiments, each collection of object representations includes one or more of object representations. In further embodiments, each collection of object representations includes or is associated with a time stamp, an order, or a time related information. In further embodiments, the first stream of collections of object representations includes one or more collections of representations of objects detected by the sensor over time. In further embodiments, the new stream of collections of object representations includes one or more collections of representations of objects detected by the sensor over time. In further embodiments, the first or the new stream of collections of object representations includes one or more collections of representations of objects in the device's surrounding. In further embodiments, the first or the new stream of collections of object representations includes one or more collections of representations of objects in a remote device's surrounding. In further embodiments, an object representation of a stream of collections of object representations includes one or more object properties. In further embodiments, the first or the new stream of collections of object representations includes one or more object properties. In further embodiments, the first stream of collections of object representations includes a comparative stream of collections of object representations whose at least one portion can be used for comparisons with at least one portion of streams of collections of object representations subsequent to the first stream of collections of object representations, the streams of collections of object representations subsequent to the first stream of collections of object representations comprising the new stream of collections of object representations. In further embodiments, the first stream of collections of object representations includes a comparative stream of collections of object representations that can be used for comparisons with the new stream of collections of object representations. In further embodiments, the new stream of collections of object representations includes an anticipatory stream of collections of object representations whose correlated one or more instruction sets can be used for anticipation of one or more instruction sets to be executed by the processor circuit.

In some embodiments, the first one or more instruction sets for operating the device include one or more instruction sets that temporally correspond to the first stream of collections of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first stream of collections of object representations include one or more instruction sets executed at a time of generating the first stream of collections of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first stream of collections of object representations include one or more instruction sets executed prior to generating

the first stream of collections of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first stream of collections of object representations include one or more instruction sets executed within a threshold period of time prior to generating the first stream of collections of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first stream of collections of object representations include one or more instruction sets executed subsequent to generating the first stream of collections of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first stream of collections of object representations include one or more instruction sets executed within a threshold period of time subsequent to generating the first stream of collections of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first stream of collections of object representations include one or more instruction sets executed within a threshold period of time prior to generating the first stream of collections of object representations or a threshold period of time subsequent to generating the first stream of collections of object representations.

In certain embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device include a knowledge of how the device operated in a circumstance. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device are included in a neuron, a node, a vertex, or an element of a knowledgebase. In further embodiments, the knowledgebase includes a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. In further embodiments, some of the neurons, nodes, vertices, or elements are interconnected. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device are structured into a knowledge cell. In further embodiments, the knowledge cell is included in a neuron, a node, a vertex, or an element of a knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device includes correlating the first stream of collections of object representations with the first one or more instruction sets for operating the device. In further embodiments, the correlating the first stream of collections of object representations with the first one or more instruction sets for operating the device includes generating a knowledge cell, the knowledge cell comprising the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device. In further embodiments, the correlating the first stream of collections of object representations with the first one or more instruction sets for operating the device includes structuring a knowledge of how the device operated in a circumstance. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device includes learning a user's knowledge, style, or methodology of operating the device in a circumstance.

In some embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device includes storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device into the memory unit, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device being part of a plurality of streams of collections of object representations correlated with one or

more instruction sets for operating the device stored in the memory unit. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device are organized into a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, each stream of collections of object representations correlated with one or more instruction sets for operating the device of the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device is included in a neuron, a node, a vertex, or an element of a knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device include a user's knowledge, style, or methodology of operating the device in circumstances. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device are stored on a remote computing device or a remote computing system. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device include an artificial intelligence system for knowledge structuring, storing, or representation. The artificial intelligence system for knowledge structuring, storing, or representation may include at least one of: a deep learning system, a supervised learning system, an unsupervised learning system, a neural network, a search-based system, an optimization-based system, a logic-based system, a fuzzy logic-based system, a tree-based system, a graph-based system, a hierarchical system, a symbolic system, a sub-symbolic system, an evolutionary system, a genetic system, a multi-agent system, a deterministic system, a probabilistic system, or a statistical system.

In certain embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes comparing at least one portion of the new stream of collections of object representations with at least one portion of the first stream of collections of object representations. In further embodiments, the at least one portion of the new stream of collections of object representations include at least one collection of object representations, at least one object representation, or at least one object property of the new stream of collections of object representations. In further embodiments, the at least one portion of the first stream of collections of object representations include at least one collection of object representations, at least one object representation, or at least one object property of the first stream of collections of object representations. In further embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes comparing at least one collection of object representations from the new stream of collections of object representations with at least one collection of object representations from the first stream of collections of object representations. In further

embodiments, the comparing at least one collection of object representations from the new stream of collections of object representations with at least one collection of object representations from the first stream of collections of object representations includes comparing at least one object representation of the at least one collection of object representations from the new stream of collections of object representations with at least one object representation of the at least one collection of object representations from the first stream of collections of object representations. The comparing at least one object representation of the at least one collection of object representations from the new stream of collections of object representations with at least one object representation of the at least one collection of object representations from the first stream of collections of object representations may include comparing at least one object property of the at least one object representation of the at least one collection of object representations from the new stream of collections of object representations with at least one object property of the at least one object representation of the at least one collection of object representations from the first stream of collections of object representations.

In some embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that there is at least a partial match between one or more portions of the new stream of collections of object representations and one or more portions of the first stream of collections of object representations. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that a similarity between at least one portion of the new stream of collections of object representations and at least one portion of the first stream of collections of object representations exceeds a similarity threshold. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining a substantial similarity between at least one portion of the new stream of collections of object representations and at least one portion of the first stream of collections of object representations. The substantial similarity may be achieved when a similarity between the at least one portion of the new stream of collections of object representations and the at least one portion of the first stream of collections of object representations exceeds a similarity threshold. The substantial similarity may be achieved when a number or a percentage of matching or partially matching portions of the new stream of collections of object representations and portions of the first stream of collections of object representations exceeds a threshold number or threshold percentage. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that a number or a percentage of matching or partially matching collections of object representations from the new stream of collections of object representations and from the first stream of collections of object representations exceeds a threshold number or threshold percentage. The matching or partially matching collections of object representations from the new stream of collections of object representations and from the first stream of collections of object representations

may be determined factoring in at least one of: an importance of a collection of object representations, an order of a collection of object representations, a threshold for a similarity in a collection of object representations, or a threshold for a difference in a collection of object representations. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that a number or a percentage of matching or partially matching object representations from the new stream of collections of object representations and from the first stream of collections of object representations exceeds a threshold number or threshold percentage. The matching or partially matching object representations from the new stream of collections of object representations and from the first stream of collections of object representations may be determined factoring in at least one of: an association of an object representation with a collection of object representations, a type of an object representation, an importance of an object representation, a threshold for a similarity in an object representation, or a threshold for a difference in an object representation. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that a number or a percentage of matching or partially matching object properties from the new stream of collections of object representations and from the first stream of collections of object representations exceeds a threshold number or threshold percentage. The matching or partially matching object properties from the new stream of collections of object representations and from the first stream of collections of object representations may be determined factoring in at least one of: an association of an object property with an object representation, an association of an object property with a collection of object representations, a category of an object property, an importance of an object property, a threshold for a similarity in an object property, or a threshold for a difference in an object property. In further embodiments, determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that there is at least a partial match between at least one collection of object representations from the new stream of collections of object representations and at least one collection of object representations from the first stream of collections of object representations. The determining that there is at least a partial match between at least one collection of object representations from the new stream of collections of object representations and at least one collection of object representations from the first stream of collections of object representations may include determining that there is at least a partial match between at least one object representation of the at least one collection of object representations from the new stream of collections of object representations and at least one object representation of the at least one collection of object representations from the first stream of collections of object representations. The determining that there is at least a partial match between at least one object representation of the at least one collection of object representations from the new stream of collections of object representations and at least one object representation of the at least one collection of object representations from the first stream of collections of object representations may include determining that there is at least a partial match between at least one object property of the at least one object representation of the at least one collection of object representations from the new stream of collections of object representations and at least one object property of the at least one object representation of the at least one collection of object representations from the first stream of collections of object representations.

In some embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations instead of or prior to an instruction set that would have been executed next. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations into a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting the processor circuit to the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting the processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes transmitting, to the processor circuit for execution, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes issuing an interrupt to the processor circuit and executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations following the interrupt. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying an element that is part of, operating on, or coupled to the processor circuit.

In certain embodiments, the processor circuit includes a logic circuit, and wherein the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the logic circuit includes a microcontroller. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying an element of the logic circuit. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes inserting the first one or more instruction sets for operating the device

correlated with the first stream of collections of object representations into an element of the logic circuit. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting the logic circuit to the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes replacing inputs into the logic circuit with the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes replacing outputs from the logic circuit with the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In some embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes causing an application for operating the device to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the application running on the processor circuit.

In certain embodiments, the system further comprises: an application including instruction sets for operating the device, the application running on the processor circuit, wherein the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying the application.

In some embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting an application to the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting an application to one or more alternate instruction sets, the application running on the processor circuit, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying at least one of: the memory unit, a register of the processor circuit, a storage, or a repository where instruction sets are stored or used. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of

object representations includes modifying one or more instruction sets for operating an application or an object of the application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying at least one of: an element of the processor circuit, an element of the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing an assembly language. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes adding or inserting additional code into a code of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object

representations includes at least one of: modifying, removing, rewriting, or overwriting a code of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of an application, the application running on the processor circuit. The branching or redirecting the code may include inserting at least one of: a branch, a jump, or a means for redirecting an execution. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes implementing a user's knowledge, style, or methodology of operating the device in a circumstance.

In certain embodiments, the system further comprises: an interface configured to cause execution of instruction sets, wherein the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations is caused by the interface. The interface may include a modification interface.

In some embodiments, the performing the one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes implementing a user's knowledge, style, or methodology of operating the device in a circumstance.

In certain embodiments, the system further comprises: an application running on the processor circuit.

In some embodiments, the instruction sets for operating the device are part of an application for operating the device, the application running on the processor circuit.

In certain embodiments, the system further comprises: an application for operating the device, the application running on the processor circuit. The application for operating the device may include the instruction sets for operating the device.

In some embodiments, the artificial intelligence unit is further configured to: receive at least one extra information. In further embodiments, the at least one extra information include one or more of: a time information, a location information, a computed information, or a contextual information. In further embodiments, the at least one extra information include one or more of: an information on an object, an information on an object representation, an information on a collection of object representations, an information on a stream of collections of object representations, an information on a device's circumstance, an information on an instruction set, an information on an application, an information on the processor circuit, an information on the device, or an information on an user. In further embodiments, the artificial intelligence unit is further configured to: learn the first stream of collections of object representations correlated with the at least one extra information. The learning the first stream of collections of object representations correlated with at least one extra information may include correlating the first stream of collections of object representations with the at least one extra information. The learning the first stream of collections of object representations correlated with at least one extra information may include storing the first stream of collections of object representations correlated with the at least one extra information into the memory unit. In further embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes anticipating the first one or more instruction sets for operating the device correlated with the first stream of

collections of object representations based on at least a partial match between an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations. The anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations may include comparing an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations. The anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations may include determining that a similarity between an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations exceeds a similarity threshold.

In certain embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: cause the user interface to present a user with an option to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In some embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: receive, via the user interface, a user's selection to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In certain embodiments, the artificial intelligence unit is further configured to: rate the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. The rating the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations may include causing a user interface to display the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations along with one or more rating values as options to be selected by a user. The rating the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations may include rating the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations without a user input.

In some embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: cause the user interface to present a user with an option to cancel the execution of the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the canceling the execution of the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes restoring the processor circuit or the device to a prior state. The restoring the processor circuit or the device to a prior state may include saving the state of the processor circuit or the device prior to executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In certain embodiments, the system further comprises: an input device configured to receive a user's operating directions, the user's operating directions for instructing the processor circuit on how to operate the device.

In some embodiments, the autonomous device operating includes a partially or a fully autonomous device operating. The partially autonomous device operating may include executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations responsive to a user confirmation. The fully autonomous device operating may include executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations without a user confirmation.

In certain embodiments, the artificial intelligence unit is further configured to: receive a second stream of collections of object representations, the second stream of collections of object representations including one or more collections of representations of objects detected by the sensor; receive a second one or more instruction sets for operating the device; and learn the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device. In further embodiments, the second stream of collections of object representations includes one or more collections of representations of objects detected by the sensor over time. In further embodiments, the second stream of collections of object representations includes or is associated with a time stamp, an order, or a time related information. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device include creating a connection between the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device. The connection may include or is associated with at least one of: an occurrence count, a weight, a parameter, or a data. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device include updating a connection between the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device. In further embodiments, the updating the connection between the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device includes updating at least one of: an occurrence count, a weight, a parameter, or a data included in or associated with the connection. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device includes storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device into a first node of a knowledgebase, and wherein the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device includes storing the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device into a second node of the knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. In further embodiments, the knowledgebase may be stored in the memory unit. The learning the first stream of collections of

object representations correlated with the first one or more instruction sets for operating the device and the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device may include creating a connection between the first node and the second node. The learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device may include updating a connection between the first node and the second node. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device is stored into a first node of a neural network and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device is stored into a second node of the neural network. The first node and the second node may be connected by a connection. The first node may be part of a first layer of the neural network and the second node may be part of a second layer of the neural network. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device is stored into a first node of a graph and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device is stored into a second node of the graph. The first node and the second node may be connected by a connection. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device is stored into a first node of a sequence and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device is stored into a second node of the sequence.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first stream of collections of object representations, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The operations may further comprise: receiving a first one or more instruction sets for operating a device. The operations may further comprise: learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device. The operations may further comprise: receiving a new stream of collections of object representations, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The operations may further comprise: anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The operations may further comprise: causing an execution of the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device

correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the execution.

In certain embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from the one or more processor circuits or from another one or more processor circuits. In further embodiments, the execution of the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations is performed by the one or more processor circuits or by another one or more processor circuits.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first stream of collections of object representations by a processor circuit, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more instruction sets for operating a device by the processor circuit. The method may further comprise: (c) learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the processor circuit. The method may further comprise: (d) receiving a new stream of collections of object representations by the processor circuit, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The method may further comprise: (e) anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, the anticipating of (e) performed by the processor circuit. The method may further comprise: (f) executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the executing of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the executing of (f).

In some embodiments, the receiving of (b) includes receiving the first one or more instruction sets for operating the device from the processor circuit or from another processor circuit. In further embodiments, the executing of (f) is performed by the processor circuit or by another processor circuit.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable as well as the following embodiments.

In certain embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device includes storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device into a memory unit, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device being part of a plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device stored in the memory unit.

In some embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations instead of or prior to an instruction set that would have been executed next. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying a register or an element of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations into a register or an element of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting a processor circuit to the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting a processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes transmitting, to a processor circuit for execution, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes issuing an interrupt to a processor circuit and executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations following the interrupt. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying an element that is part of, operating on, or coupled to a processor circuit.

In certain embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes executing, by a logic circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the logic circuit includes a microcontroller. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying an element of the logic circuit. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations into an element of the logic circuit. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting the logic circuit to the first one or more instruction sets for operating the device correlated with

the first stream of collections of object representations. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes replacing inputs into the logic circuit with the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes replacing outputs from the logic circuit with the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In some embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes executing, by an application for operating the device, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying an application, the application including instruction sets for operating the device. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting an application to the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting an application to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying at least one of: the memory unit, a register of a processor circuit, a storage, or a repository where instruction sets are stored or used. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets for operating an application or an object of the application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying at least one of: an element of a processor circuit, an element of the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further

embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing an assembly language. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes adding or inserting additional code into a code of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes at least one of: modifying, removing, rewriting, or overwriting a code of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of an application. The branching or redirecting the code may include inserting at least one of: a branch, a jump, or a means for redirecting an execution. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes implementing a user's knowledge, style, or methodology of operating the device in a circumstance. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations via an interface. The interface may include a modification interface.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving at least one extra information. In further embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: learning the first stream of collections of object representations correlated with the at least one extra information.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: presenting, via a user interface, a user with an option to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving, via a user interface, a user's selection to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: rating the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: presenting, via a user interface, a user with an option to cancel the execution of the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving, via an input device, a user's operating directions, the user's operating directions for instructing a processor circuit on how to operate the device.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving a second stream of collections of object representations, the second stream of collections of object representations including one or more collections of representations of objects detected by the sensor; receiving a second one or more instruction sets for operating the device; and learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving, by a first processor circuit of the one or more processor circuits, a first stream of collections of object representations, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The operations may further comprise: receiving, by the first processor circuit of the one or more processor circuits, a first one or more instruction sets for operating a device. The operations may further comprise: learning, by the first processor circuit of the one or more processor circuits, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device. The operations may further comprise: receiving, by the first processor circuit of the one or more processor circuits, a new stream of collections of object representations, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The operations may further comprise: anticipating, by the first processor circuit of the one or more processor circuits, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The operations may further comprise: causing, by the first processor circuit of the one or more processor circuits, an execution, by a second processor circuit of the one or more processor circuits, of the first one or more instruction sets for operating the device correlated with the first stream of collections of object

representations, the causing performed in response to the anticipating, by the first processor circuit of the one or more processor circuits, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the execution.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first stream of collections of object representations by a first processor circuit, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more instruction sets for operating a device by the first processor circuit. The method may further comprise: (c) learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the first processor circuit. The method may further comprise: (d) receiving a new stream of collections of object representations by the first processor circuit, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The method may further comprise: (e) anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, the anticipating of (e) performed by the first processor circuit. The method may further comprise: (f) executing, by a second processor circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the executing of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the executing of (f).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for learning a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first stream of collections of object representations, the first stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: learn the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first stream of collections of object representations, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The operations may further comprise: receiving a first one or more instruction sets for operating a device. The operations may further comprise: learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a non method comprising: (a) receiving a first stream of collections of object representations by a processor circuit, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more instruction sets for operating a device by the processor circuit. The method may further comprise: (c) learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the processor circuit.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: access the memory unit that comprises a plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device including a first stream of collections of object representations correlated with a first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: receive a new stream of collections of object representations, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations performed in response to the executing by the processor circuit.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits

cause the one or more processor circuits to perform operations comprising: accessing a memory unit that comprises a plurality of streams of collections of object representations correlated with one or more instruction sets for operating a device, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device including a first stream of collections of object representations correlated with a first one or more instruction sets for operating the device. The operations may further comprise: receiving a new stream of collections of object representations, the new stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The operations may further comprise: anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The operations may further comprise: causing an execution of the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the execution.

In some aspects, the disclosure relates to a method comprising: (a) accessing a memory unit that comprises a plurality of streams of collections of object representations correlated with one or more instruction sets for operating a device, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device including a first stream of collections of object representations correlated with a first one or more instruction sets for operating the device, the accessing of (a) performed by a processor circuit. The method may further comprise: (b) receiving a new stream of collections of object representations by the processor circuit, the new stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The method may further comprise: (c) anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, the anticipating of (c) performed by the processor circuit. The method may further comprise: (d) executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the executing of (d) performed in response to the anticipating of (c). The method may further comprise: (e) performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the executing of (d).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for learning and using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices.

In some embodiments, the system comprises: a logic circuit configured to receive inputs and produce outputs, wherein the outputs are used for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first collection of object representations, the first collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more inputs, wherein the first one or more inputs are also received by the logic circuit. The artificial intelligence unit may be further configured to: learn the first collection of object representations correlated with the first one or more inputs. The artificial intelligence unit may be further configured to: receive a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may be further configured to: cause the logic circuit to receive the first one or more inputs correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by one or more outputs for operating the device produced by the logic circuit.

In some embodiments, the logic circuit configured to receive inputs and produce outputs includes a logic circuit configured to produce outputs based at least in part on logic operations performed on the inputs. In further embodiments, the learning the first collection of object representations correlated with the first one or more inputs includes correlating the first collection of object representations with the first one or more inputs. In further embodiments, the learning the first collection of object representations correlated with the first one or more inputs includes storing the first collection of object representations correlated with the first one or more inputs into the memory unit, the first collection of object representations correlated with the first one or more inputs being part of a plurality of collections of object representations correlated with one or more inputs stored in the memory unit. In further embodiments, the anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes comparing at least one portion of the new collection of object representations with at least one portion of the first collection of object representations. In further embodiments, the anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes determining that there is at least a partial match between the new collection of object representations and the first collection of object representations. In further embodiments, the causing the logic circuit to receive the first one or more inputs correlated with the first collection of object representations includes transmitting, to the logic circuit, the first one or more inputs correlated with the first collection of object representations. In further embodiments, the causing the logic circuit to receive the first one or more inputs correlated with the first collection of object representations includes replacing one or more inputs into the logic circuit with the first one or more inputs correlated with the first collection of object representations.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first collection of object representations, the first collection of object representations including one or more representations of objects detected by a sensor. The operations may further comprise: receiving a first one or more inputs, wherein the first one or more inputs are also received by a logic circuit, and wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device. The operations may further comprise: learning the first collection of object representations correlated with the first one or more inputs. The operations may further comprise: receiving a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The operations may further comprise: anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further comprise: causing the logic circuit to receive the first one or more inputs correlated with the first collection of object representations, the causing performed in response to the anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, wherein the device performs one or more operations defined by one or more outputs for operating the device produced by the logic circuit.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a processor circuit, the first collection of object representations including one or more representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more inputs by the processor circuit, wherein the first one or more inputs are also received by a logic circuit, and wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device. The method may further comprise: (c) learning the first collection of object representations correlated with the first one or more inputs, the learning of (c) performed by the processor circuit. The method may further comprise: (d) receiving a new collection of object representations by the processor circuit, the new collection of object representations including one or more representations of objects detected by the sensor. The method may further comprise: (e) anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (e) performed by the processor circuit. The method may further comprise: (f) receiving, by the logic circuit, the first one or more inputs correlated with the first collection of object representations, the receiving of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the device, one or more operations defined by one or more outputs for operating the device produced by the logic circuit.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for learning and using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a logic circuit configured to receive inputs and produce outputs, wherein the outputs are used for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence. The artificial intelligence unit may be configured to: receive a first collection of object representations, the first collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more outputs, the first one or more outputs transmitted from the logic circuit. The artificial intelligence unit may be further configured to: learn the first collection of object representations correlated with the first one or more outputs. The artificial intelligence unit may be further configured to: receive a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more outputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may be further configured to: cause the device to perform one or more operations defined by the first one or more outputs correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit.

In some embodiments, the learning the first collection of object representations correlated with the first one or more outputs includes correlating the first collection of object representations with the first one or more outputs. In further embodiments, the learning the first collection of object representations correlated with the first one or more outputs includes storing the first collection of object representations correlated with the first one or more outputs into the memory unit, the first collection of object representations correlated with the first one or more outputs being part of a plurality of collections of object representations correlated with one or more outputs stored in the memory unit. In further embodiments, the anticipating the first one or more outputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes comparing at least one portion of the new collection of object representations with at least one portion of the first collection of object representations. In further embodiments, the anticipating the first one or more outputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes determining that there is at least a partial match between the new collection of object representations and the first collection of object representations. In further embodiments, the causing the device to perform one or more operations defined by the first one or more outputs correlated with the first collection of object representations includes replacing one or more outputs from the logic circuit with the first one or more outputs correlated with the first collection of object representations.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first collection of object representations, the first collection of object representations including one or more representations of objects

detected by a sensor. The operations may further comprise: receiving a first one or more outputs, the first one or more outputs transmitted from a logic circuit, wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device. The operations may further comprise: learning the first collection of object representations correlated with the first one or more outputs. The operations may further comprise: receiving a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The operations may further comprise: anticipating the first one or more outputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further comprise: causing the device to perform one or more operations defined by the first one or more outputs correlated with the first collection of object representations, the causing performed in response to the anticipating the first one or more outputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a processor circuit, the first collection of object representations including one or more representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more outputs by the processor circuit, the first one or more outputs transmitted from a logic circuit, wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device. The method may further comprise: (c) learning the first collection of object representations correlated with the first one or more outputs, the learning of (c) performed by the processor circuit. The method may further comprise: (d) receiving a new collection of object representations by the processor circuit, the new collection of object representations including one or more representations of objects detected by the sensor. The method may further comprise: (e) anticipating the first one or more outputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (e) performed by the processor circuit. The method may further comprise: (f) performing, by the device, one or more operations defined by the first one or more outputs correlated with the first collection of object representations, the one or more operations by the device performed in response to the anticipating of (e).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for learning and using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: an actuator configured to receive inputs and perform motions. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first collection of object representations, the first collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence

unit may be further configured to: receive a first one or more inputs, wherein the first one or more inputs are also received by the actuator. The artificial intelligence unit may be further configured to: learn the first collection of object representations correlated with the first one or more inputs. The artificial intelligence unit may be further configured to: receive a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may be further configured to: cause the actuator to receive the first one or more inputs correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the actuator performs one or more motions defined by the first one or more inputs correlated with the first collection of object representations.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first collection of object representations, the first collection of object representations including one or more representations of objects detected by a sensor. The operations may further comprise: receiving a first one or more inputs, wherein the first one or more inputs are also received by an actuator, and wherein the actuator is configured to receive inputs and perform motions. The operations may further comprise: learning the first collection of object representations correlated with the first one or more inputs. The operations may further comprise: receiving a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The operations may further comprise: anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further comprise: causing the actuator to receive the first one or more inputs correlated with the first collection of object representations, the causing performed in response to the anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, wherein the actuator performs one or more motions defined by the first one or more inputs correlated with the first collection of object representations.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a processor circuit, the first collection of object representations including one or more representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more inputs by the processor circuit, wherein the first one or more inputs are also received by an actuator, and wherein the actuator is configured to receive inputs and perform motions. The method may further comprise: (c) learning the first collection of object representations correlated with the first one or more inputs, the learning of (c) performed by the processor circuit. The method may further comprise: (d) receiving a new collection of object representations by the processor circuit, the new collection of object representations including one or more representations of objects detected by the sensor. The method may further comprise: (e) anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (e) performed by the

processor circuit. The method may further comprise: (f) receiving, by the actuator, the first one or more inputs correlated with the first collection of object representations, the receiving of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the actuator, one or more motions defined by the first one or more inputs correlated with the first collection of object representations.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

Other features and advantages of the disclosure will become apparent from the following description, including the claims and drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

Fig. 1 illustrates a block diagram of Computing Device 70 that can provide processing capabilities used in some of the disclosed embodiments.

Fig. 2 illustrates an embodiment of Device 98 comprising Unit for Learning and/or Using a Device's Circumstances for Autonomous Device Operation (DCADO Unit 100).

Figs. 3A-3E illustrate various embodiments of Sensors 92 and elements of Object Processing Unit 93.

Figs. 4A-4B, illustrate an exemplary embodiment of Objects 615 detected in Device's 98 surrounding, and resulting Collection of Object Representations 525.

Fig. 5 illustrates some embodiments of obtaining instruction sets, data, and/or other information through tracing, profiling, or sampling of Processor 11 registers, memory, or other computing system components.

Figs. 6A-6B illustrate some embodiments of obtaining instruction sets, data, and/or other information through tracing, profiling, or sampling of Logic Circuit 250.

Figs. 7A-7E illustrate some embodiments of Instruction Sets 526.

Figs. 8A-8B illustrate some embodiments of Extra Information 527.

Fig. 9 illustrates an embodiment where DCADO Unit 100 is part of or operating on Processor 11.

Fig. 10 illustrates an embodiment where DCADO Unit 100 resides on Server 96 accessible over Network 95.

Fig. 11 illustrates an embodiment of learning and/or using Remote Device's 97 circumstances for autonomous Remote Device 97 operation.

Fig. 12 illustrates an embodiment of Artificial Intelligence Unit 110.

Fig. 13 illustrates an embodiment of Knowledge Structuring Unit 520 correlating individual Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527.

Fig. 14 illustrates another embodiment of Knowledge Structuring Unit 520 correlating individual Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527.

Fig. 15 illustrates an embodiment of Knowledge Structuring Unit 520 correlating streams of Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527.

Fig. 16 illustrates another embodiment of Knowledge Structuring Unit 520 correlating streams of Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527.

Fig. 17 illustrates various artificial intelligence methods, systems, and/or models that can be utilized in DCADO Unit 100 embodiments.

Fig. 18A-18C illustrate embodiments of interconnected Knowledge Cells 800 and updating weights of Connections 853.

Fig. 19 illustrates an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Collection of Knowledge Cells 530d.

Fig. 20 illustrates an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Neural Network 530a.

Fig. 21 illustrates an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Neural Network 530a comprising shortcut Connections 853.

Fig. 22 illustrates an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Graph 530b.

Fig. 23 illustrates an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Collection of Sequences 530c.

Fig. 24 illustrates an embodiment of determining anticipatory Instruction Sets 526 from a single Knowledge Cell 800.

Fig. 25 illustrates an embodiment of determining anticipatory Instruction Sets 526 by traversing a single Knowledge Cell 800.

Fig. 26 illustrates an embodiment of determining anticipatory Instruction Sets 526 using collective similarity comparisons.

Fig. 27 illustrates an embodiment of determining anticipatory Instruction Sets 526 using Neural Network 530a.

Fig. 28 illustrates an embodiment of determining anticipatory Instruction Sets 526 using Graph 530b.

Fig. 29 illustrates an embodiment of determining anticipatory Instruction Sets 526 using Collection of Sequences 530c.

Fig. 30 illustrates some embodiments of modifying execution and/or functionality of Processor 11 through modification of Processor 11 registers, memory, or other computing system components.

Figs. 31A-31B illustrate some embodiments of modifying execution and/or functionality of Logic Circuit 250 through modification of inputs and/or outputs of Logic Circuit 250.

Fig. 32 illustrates a flow chart diagram of an embodiment of method 9100 for learning and/or using a device's circumstances for autonomous device operation.

Fig. 33 illustrates a flow chart diagram of an embodiment of method 9200 for learning and/or using a device's circumstances for autonomous device operation.

Fig. 34 illustrates a flow chart diagram of an embodiment of method 9300 for learning and/or using a device's circumstances for autonomous device operation.

Fig. 35 illustrates a flow chart diagram of an embodiment of method 9400 for learning and/or using a device's circumstances for autonomous device operation.

Fig. 36 illustrates a flow chart diagram of an embodiment of method 9500 for learning and/or using a device's circumstances for autonomous device operation.

Fig. 37 illustrates a flow chart diagram of an embodiment of method 9600 for learning and/or using a device's circumstances for autonomous device operation.

Fig. 38 illustrates an exemplary embodiment of Loader 98a.

Fig. 39 illustrates an exemplary embodiment of Boat 98b.

Fig. 40 illustrates an exemplary embodiment of utilizing Area of Interest 450 around Boat 98b.

Like reference numerals in different figures indicate like elements. Horizontal or vertical "…" or other such indicia may be used to indicate additional instances of the same type of element. n, m, x, or other such letters or indicia represent integers or other sequential numbers that follow the sequence where they are indicated. It should be noted that n, m, x, or other such letters or indicia may represent different numbers in different elements even where the elements are depicted in the same figure. In general, n, m, x, or other such letters or indicia may follow the sequence and/or context where they are indicated. Any of these or other such letters or indicia may be used interchangeably depending on context and space available. The drawings are not necessarily to scale, with emphasis instead being placed upon illustrating the embodiments, principles, and concepts of the disclosure. A line or arrow between any of the disclosed elements comprises an interface that enables the coupling, connection, and/or interaction between the elements.

DETAILED DESCRIPTION

The disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation comprise apparatuses, systems, methods, features, functionalities, and/or applications that enable learning a device's circumstances including objects with various properties along with correlated instruction sets for operating the device, storing this knowledge in a knowledgebase (i.e. neural network, graph, sequences, etc.), and operating a device autonomously. The disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, any of their elements, any of their embodiments, or a combination thereof can generally be referred to as DCADO, DCADO Unit, or as other suitable name or reference.

Referring now to Fig. 1, an embodiment is illustrated of Computing Device 70 (also referred to simply as computing device, computing system, or other suitable name or reference, etc.) that can provide processing capabilities used in some embodiments of the forthcoming disclosure. Later described devices, systems, and methods, in combination with processing capabilities of Computing Device 70, enable learning and/or using a device's circumstances for autonomous device operation and/or other functionalities described herein. Various embodiments of the disclosed devices, systems, and methods include hardware, functions, logic, programs, and/or a combination thereof that can be implemented using any type or form of computing, computing enabled, or other device or system such as a mobile device, a computer, a computing enabled telephone, a server, a gaming device, a television device, a digital camera, a GPS receiver, a media player, an embedded device, a supercomputer, a wearable device, an implantable device, a cloud, or any other type or form of computing, computing enabled, or

other device or system capable of performing the operations described herein.

In some designs, Computing Device 70 comprises hardware, processing techniques or capabilities, programs, or a combination thereof. Computing Device 70 includes one or more central processing units, which may also be referred to as processors 11. Processor 11 includes one or more memory ports 10 and/or one or more input-output ports, also referred to as I/O ports 15, such as I/O ports 15A and 15B. Processor 11 may be special or general purpose. Computing Device 70 may further include memory 12, which can be connected to the remainder of the components of Computing Device 70 via bus 5. Memory 12 can be connected to processor 11 via memory port 10. Computing Device 70 may also include display device 21 such as a monitor, projector, glasses, and/or other display device. Computing Device 70 may also include Human-machine Interface 23 such as a keyboard, a pointing device, a mouse, a touchscreen, a joystick, a remote controller, and/or other input device. In some implementations, Human-machine Interface 23 can be connected with bus 5 or directly connected with specific elements of Computing Device 70. Computing Device 70 may include additional elements such as one or more input/output devices 13. Processor 11 may include or be interfaced with cache memory 14. Storage 27 may include memory, which provides an operating system 17 (i.e. also referred to as OS 17, etc.), additional application programs 18, and/or data space 19 in which additional data or information can be stored. Alternative memory device 16 can be connected to the remaining components of Computing Device 70 via bus 5. Network interface 25 can also be connected with bus 5 and be used to communicate with external computing devices via a network. Some or all described elements of Computing Device 70 can be directly or operatively connected or coupled with each other using any other connection means known in art. Other additional elements may be included as needed, or some of the disclosed ones may be excluded, or a combination thereof may be utilized in alternate implementations of Computing Device 70.

Processor 11 includes one or more circuits or devices that can execute instructions fetched from memory 12 and/or other element. Processor 11 may include any combination of hardware and/or processing techniques or capabilities for executing or implementing logic functions or programs. Processor 11 may include a single core or a multi core processor. Processor 11 includes the functionality for loading operating system 17 and operating any application programs 18 thereon. In some embodiments, Processor 11 can be provided in a microprocessing or a processing unit, such as, for example, Snapdragon processor produced by Qualcomm Inc., processor by Intel Corporation of Mountain View, California, processor manufactured by Motorola Corporation of Schaumburg, Ill.; processor manufactured by Transmeta Corporation of Santa Clara, Calif.; processor manufactured by International Business Machines of White Plains, N.Y.; processor manufactured by Advanced Micro Devices of Sunnyvale, California, or any computing circuit or device for performing similar functions. In other embodiments, processor 11 can be provided in a graphics processing unit (GPU), visual processing unit (VPU), or other highly parallel processing circuit or device such as, for example, nVidia GeForce line of GPUs, AMD Radeon line of GPUs, and/or others. Such GPUs or other highly parallel processing circuits or devices may provide superior performance in processing operations on neural networks, graphs, and/or other data structures. In further embodiments, processor 11 can be provided in a micro controller such as, for example, Texas instruments, Atmel, Microchip Technology, ARM, Silicon Labs, Intel, and/or other lines of micro controllers. In further embodiments, processor 11 can be provided in a quantum processor such as, for example, D-Wave Systems, Microsoft, Intel, IBM, Google, Toshiba, and/or other lines of quantum processors. In further embodiments, processor 11 can be provided in a biocomputer

such as DNA-based computer, protein-based computer, molecule-based computer, and/or others. In further embodiments, processor 11 includes any circuit or device for performing logic operations. Processor 11 can be based on any of the aforementioned or other available processors capable of operating as described herein. Computing Device 70 may include one or more of the aforementioned or other processors. In some designs, processor 11 can communicate with memory 12 via a system bus 5. In other designs, processor 11 can communicate directly with memory 12 via a memory port 10.

Memory 12 includes one or more circuits or devices capable of storing data. In some embodiments, Memory 12 can be provided in a semiconductor or electronic memory chip such as static random access memory (SRAM), Flash memory, Burst SRAM or SynchBurst SRAM (BSRAM), Dynamic random access memory (DRAM), Fast Page Mode DRAM (FPM DRAM), Enhanced DRAM (EDRAM), Extended Data Output RAM (EDO RAM), Extended Data Output DRAM (EDO DRAM), Burst Extended Data Output DRAM (BEDO DRAM), Enhanced DRAM (EDRAM), synchronous DRAM (SDRAM), JEDEC SRAM, PC100 SDRAM, Double Data Rate SDRAM (DDR SDRAM), Enhanced SDRAM (ESDRAM), SyncLink DRAM (SLDRAM), Direct Rambus DRAM (DRDRAM), Ferroelectric RAM (FRAM), and/or others. In other embodiments, Memory 12 includes any volatile memory. In general, Memory 12 can be based on any of the aforementioned or other available memories capable of operating as described herein.

Storage 27 includes one or more devices or mediums capable of storing data. In some embodiments, Storage 27 can be provided in a device or medium such as a hard drive, flash drive, optical disk, and/or others. In other embodiments, Storage 27 can be provided in a biological storage device such as DNA-based storage device, protein-based storage device, molecule-based storage device, and/or others. In further embodiments, Storage 27 can be provided in an optical storage device such as holographic storage, and/or others. In further embodiments, Storage 27 may include any non-volatile memory. In general, Storage 27 can be based on any of the aforementioned or other available storage devices or mediums capable of operating as described herein. In some aspects, Storage 27 may include any features, functionalities, and embodiments of Memory 12, and vice versa, as applicable.

Processor 11 can communicate directly with cache memory 14 via a connection means such as a secondary bus which may also sometimes be referred to as a backside bus. In some embodiments, processor 11 can communicate with cache memory 14 using the system bus 5. Cache memory 14 may typically have a faster response time than main memory 12 and can include a type of memory which is considered faster than main memory 12 such as, for example, SRAM, BSRAM, or EDRAM. Cache memory includes any structure such as multilevel caches, for example. In some embodiments, processor 11 can communicate with one or more I/O devices 13 via a system bus 5. Various busses can be used to connect processor 11 to any of the I/O devices 13 such as a VESA VL bus, an ISA bus, an EISA bus, a MicroChannel Architecture (MCA) bus, a PCI bus, a PCI-X bus, a PCI-Express bus, a NuBus, and/or others. In some embodiments, processor 11 can communicate directly with I/O device 13 via HyperTransport, Rapid I/O, or InfiniBand. In further embodiments, local busses and direct communication can be mixed. For example, processor 11 can communicate with an I/O device 13 using a local interconnect bus and communicate with another I/O device 13 directly. Similar configurations can be used for any other components described herein.

Computing Device 70 may further include alternative memory such as a SD memory slot, a USB memory stick, an optical drive such as a CD-ROM drive, a CD-R/RW drive, a DVD-ROM drive or a BlueRay disc, a hard-drive, and/or any other device comprising non-volatile memory suitable for storing data or installing application programs. Computing Device 70 may further include a storage device 27 comprising any type or form of non-volatile memory for storing an operating system (OS) such as any type or form of Windows OS, Mac OS, Unix OS, Linux OS, Android OS, iPhone OS, mobile version of Windows OS, an embedded OS, or any other OS that can operate on Computing Device 70. Computing Device 70 may also include application programs 18, and/or data space 19 for storing additional data or information. In some embodiments, alternative memory 16 can be used as or similar to storage device 27. Additionally, OS 17 and/or application programs 18 can be operable from a bootable medium such as, for example, a flash drive, a micro SD card, a bootable CD or DVD, and/or other bootable medium.

Application Program 18 (also referred to as program, computer program, application, script, code, or other suitable name or reference) comprises instructions that can provide functionality when executed by processor 11. As such, Application Program 18 may be used to operate (i.e. perform operations on/with) or control a device or system. Application program 18 can be implemented in a high-level procedural or object-oriented programming language, or in a low-level machine or assembly language. Any language used can be compiled, interpreted, or otherwise translated into machine language. Application program 18 can be deployed in any form including as a stand-alone program or as a module, component, subroutine, or other unit suitable for use in a computing system. Application program 18 does not necessarily correspond to a file in a file system. A program can be stored in a portion of a file that may hold other programs or data, in a single file dedicated to the program, or in multiple files (i.e. files that store one or more modules, sub programs, or portions of code, etc.). Application Program 18 can be delivered in various forms such as, for example, executable file, library, script, plugin, addon, applet, interface, console application, web application, application service provider (ASP)-type application, operating system, and/or other forms. Application program 18 can be deployed to be executed on one computing device or on multiple computing devices (i.e. cloud, distributed, or parallel computing, etc.), or at one site or distributed across multiple sites interconnected by a communication network or an interface.

Network interface 25 can be utilized for interfacing Computing Device 70 with other devices via a network through a variety of connections including telephone lines, wired or wireless connections, LAN or WAN links (i.e. 802.11, T1, T3, 56 kb, X.25, etc.), broadband connections (i.e. ISDN, Frame Relay, ATM, etc.), or a combination thereof. Examples of networks include the Internet, an intranet, an extranet, a local area network (LAN), a wide area network (WAN), a personal area network (PAN), a home area network (HAN), a campus area network (CAN), a metropolitan area network (MAN), a global area network (GAN), a storage area network (SAN), virtual network, a virtual private network (VPN), a Bluetooth network, a wireless network, a wireless LAN, a radio network, a HomePNA, a power line communication network, a G.hn network, an optical fiber network, an Ethernet network, an active networking network, a client-server network, a peer-to-peer network, a bus network, a star network, a ring network, a mesh network, a star-bus network, a tree network, a hierarchical topology network, and/or other networks. Network interface 25 may include a built-in network adapter, network interface card, PCMCIA network card, card bus network adapter, wireless network adapter, Bluetooth network adapter, WiFi network adapter, USB network adapter, modem, and/or any other device suitable for interfacing Computing Device 70 with any type of network capable of communication and/or operations described herein.

I/O devices 13 may be present in various shapes or forms in Computing Device 70. Examples of I/O device 13 capable of input include a joystick, a keyboard, a mouse, a trackpad, a trackpoint, a touchscreen, a trackball, a microphone, a drawing tablet, a glove, a tactile input device, a still or video camera, and/or other input device. Examples of I/O device 13 capable of output include a video display, a touchscreen, a projector, a glasses, a speaker, a tactile output device, and/or other output device. Examples of I/O device 13 capable of input and output include a disk drive, an optical storage device, a modem, a network card, and/or other input/output device. I/O device 13 can be interfaced with processor 11 via an I/O port 15, for example. In some aspects, I/O device 13 can be a bridge between system bus 5 and an external communication bus such as a USB bus, an Apple Desktop Bus, an RS-232 serial connection, a SCSI bus, a FireWire bus, a FireWire 800 bus, an Ethernet bus, an AppleTalk bus, a Gigabit Ethernet bus, an Asynchronous Transfer Mode bus, a HIPPI bus, a Super HIPPI bus, a SerialPlus bus, a SCI/LAMP bus, a FibreChannel bus, a Serial Attached small computer system interface bus, and/or other bus.

An output interface (not shown) such as a graphical user interface, an acoustic output interface, a tactile output interface, any device driver (i.e. audio, video, or other driver), and/or other output interface or system can be utilized to process output from elements of Computing Device 70 for conveyance on an output device such as Display 21. In some aspects, Display 21 or other output device itself may include an output interface for processing output from elements of Computing Device 70. Further, an input interface (not shown) such as a keyboard listener, a touchscreen listener, a mouse listener, any device driver (i.e. audio, video, keyboard, mouse, touchscreen, or other driver), and/or other input interface or system can be utilized to process input from Human-machine Interface 23 or other input device for use by elements of Computing Device 70. In some aspects, Human-machine Interface 23 or other input device itself may include an input interface for processing input for use by elements of Computing Device 70.

Computing Device 70 may include or be connected to multiple display devices 21. Display devices 21 can each be of the same or different type or form. Computing Device 70 and/or its elements comprise any type or form of suitable hardware, programs, or a combination thereof to support, enable, or provide for the connection and use of multiple display devices 21. In one example, Computing Device 70 includes any type or form of video adapter, video card, driver, and/or library to interface, communicate, connect, or otherwise use display devices 21. In some aspects, a video adapter may include multiple connectors to interface to multiple display devices 21. In other aspects, Computing Device 70 includes multiple video adapters, with each video adapter connected to one or more display devices 21. In some embodiments, Computing Device's 70 operating system can be configured for using multiple displays 21. In other embodiments, one or more display devices 21 can be provided by one or more other computing devices such as remote computing devices connected to Computing Device 70 via a network or an interface.

Computing Device 70 can operate under the control of operating system 17, which may support Computing Device's 70 basic functions, interface with and manage hardware resources, interface with and manage peripherals, provide common services for application programs, schedule tasks, and/or perform other functionalities. A modern operating system enables features and functionalities such as a high resolution display, graphical user interface (GUI), touchscreen, cellular network connectivity (i.e. mobile operating system, etc.), Bluetooth connectivity, WiFi connectivity, global positioning system (GPS) capabilities, mobile navigation, microphone, speaker, still picture camera, video camera, voice recorder, speech recognition, music player, video player, near field communication,

personal digital assistant (PDA), and/or other features, functionalities, or applications. For example, Computing Device 70 can use any conventional operating system, any embedded operating system, any real-time operating system, any open source operating system, any video gaming operating system, any proprietary operating system, any online operating system, any operating system for mobile computing devices, or any other operating system capable of running on Computing Device 70 and performing operations described herein. Example of operating systems include Windows XP, Windows 7, Windows 8, Windows 10, etc. manufactured by Microsoft Corporation of Redmond, Wash.; Mac OS, iPhone OS, etc. manufactured by Apple Computer of Cupertino, Calif.; OS/2 manufactured by International Business Machines of Armonk, N.Y.; Linux, a freely-available operating system distributed by Caldera Corp. of Salt Lake City, Utah; or any type or form of a Unix operating system, and/or others. Any operating systems such as the ones for Android devices can similarly be utilized.

Computing Device 70 can be implemented as or be part of various model architectures such as web services, distributed computing, grid computing, cloud computing, and/or other architectures. For example, in addition to the traditional desktop, server, or mobile operating system architectures, a cloud-based operating system can be utilized to provide the structure on which embodiments of the disclosure can be implemented. Other aspects of Computing Device 70 can also be implemented in the cloud without departing from the spirit and scope of the disclosure. For example, memory, storage, processing, and/or other elements can be hosted in the cloud. In some embodiments, Computing Device 70 can be implemented on multiple devices. For example, a portion of Computing Device 70 can be implemented on a mobile device and another portion can be implemented on wearable electronics.

Computing Device 70 can be or include any mobile device, a mobile phone, a smartphone (i.e. iPhone, Windows phone, Blackberry phone, Android phone, etc.), a tablet, a personal digital assistant (PDA), wearable electronics, implantable electronics, and/or other mobile device capable of implementing the functionalities described herein. Computing Device 70 can also be or include an embedded device, which can be any device or system with a dedicated function within another device or system. Embedded systems range from the simplest ones dedicated to one task with no user interface to complex ones with advanced user interface that may resemble modern desktop computer systems. Examples of devices comprising an embedded device include a mobile telephone, a personal digital assistant (PDA), a gaming device, a media player, a digital still or video camera, a pager, a television device, a set-top box, a personal navigation device, a global positioning system (GPS) receiver, a portable storage device (i.e. a USB flash drive, etc.), a digital watch, a DVD player, a printer, a microwave oven, a washing machine, a dishwasher, a gateway, a router, a hub, an automobile entertainment system, an automobile navigation system, a refrigerator, a washing machine, a factory automation device, an assembly line device, a factory floor monitoring device, a thermostat, an automobile, a factory controller, a telephone, a network bridge, and/or other devices. An embedded device can operate under the control of an operating system for embedded devices such as MicroC/OS-II, QNX, VxWorks, eCos, TinyOS, Windows Embedded, Embedded Linux, and/or other embedded device operating systems.

Various implementations of the disclosed devices, systems, and methods can be realized in digital electronic circuitry, integrated circuitry, logic gates, application specific integrated circuits (ASICs), field programmable gate arrays (FPGAs), computer hardware, firmware, programs, virtual machines, and/or combinations thereof including their structural, logical, and/or physical equivalents.

The disclosed devices, systems, and methods may include clients and servers. A client and server are generally, but not always, remote from each other and typically, but not always, interact via a network or an interface. The relationship of a client and server may arise by virtue of computer programs running on their respective computers and having a client-server relationship to each other, for example.

The disclosed devices, systems, and methods can be implemented in a computing system that includes a back end component, a middleware component, a front end component, or any combination thereof. The components of the system can be interconnected by any form or medium of digital data communication such as, for example, a network.

Computing Device 70 may include or be interfaced with a computer program product comprising instructions or logic encoded on a computer-readable medium. Such instructions or logic, when executed, may configure or cause one or more processors to perform the operations and/or functionalities disclosed herein. For example, a computer program can be provided or encoded on a computer-readable medium such as an optical medium (i.e. DVD-ROM, etc.), flash drive, hard drive, any memory, firmware, or other medium. Computer program can be installed onto a computing device to cause the computing device to perform the operations and/or functionalities disclosed herein. Machine-readable medium, computer-readable medium, or other such terms may refer to any computer program product, apparatus, and/or device for providing instructions and/or data to one or more programmable processors. As such, machine-readable medium includes any medium that can send and/or receive machine instructions as a machine-readable signal. Examples of a machine-readable medium include a volatile and/or non-volatile medium, a removable and/or non-removable medium, a communication medium, a storage medium, and/or other medium. A communication medium, for example, can transmit computer readable instructions and/or data in a modulated data signal such as a carrier wave or other transport technique, and may include any other form of information delivery medium known in art. A non-transitory machine-readable medium comprises all machine-readable media except for a transitory, propagating signal.

In some embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, can be implemented entirely or in part in a device (i.e. microchip, circuitry, logic gates, electronic device, computing device, special or general purpose processor, etc.) or system that comprises (i.e. hard coded, internally stored, etc.) or is provided with (i.e. externally stored, etc.) instructions for implementing DCADO functionalities. As such, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, may include the processing, memory, storage, and/or other features, functionalities, and embodiments of Computing Device 70 or elements thereof. Such device or system can operate on its own (i.e. standalone device or system, etc.), be embedded in another device or system (i.e. an industrial machine, a robot, a vehicle, a toy, a smartphone, a television device, an appliance, and/or any other device or system capable of housing the elements needed for DCADO functionalities), work in combination with other devices or systems, or be available in any other configuration. In other embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, may include Alternative Memory 16 that provides instructions for implementing DCADO functionalities to one or more Processors 11. In further embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, can be

implemented entirely or in part as a computer program and executed by one or more Processors 11. Such program can be implemented in one or more modules or units of a single or multiple computer programs. Such program may be able to attach to or interface with, inspect, and/or take control of another application program to implement DCADO functionalities. In further embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, can be implemented as a network, web, distributed, cloud, or other such application accessed on one or more remote computing devices (i.e. servers, cloud, etc.) via Network Interface 25, such remote computing devices including processing capabilities and instructions for implementing DCADO functionalities. In further embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, can be (1) attached to or interfaced with any computing device or application program, (2) included as a feature of an operating system, (3) built (i.e. hard coded, etc.) into any computing device or application program, and/or (4) available in any other configuration to provide its functionalities.

In some embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, can be implemented at least in part in a computer program such as Java application or program. Java provides a robust and flexible environment for application programs including flexible user interfaces, robust security, built-in network protocols, powerful application programming interfaces, database or DBMS connectivity and interfacing functionalities, file manipulation capabilities, support for networked applications, and/or other features or functionalities. Application programs based on Java can be portable across many devices, yet leverage each device's native capabilities. Java supports the feature sets of most smartphones and a broad range of connected devices while still fitting within their resource constraints. Various Java platforms include virtual machine features comprising a runtime environment for application programs. Java platforms provide a wide range of user-level functionalities that can be implemented in application programs such as displaying text and graphics, playing and recording audio content, displaying and recording visual content, communicating with another computing device, and/or other functionalities. It should be understood that the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, are programming language, platform, and operating system independent. Examples of programming languages that can be used instead of or in addition to Java include C, C++, Cobol, Python, Java Script, Tcl, Visual Basic, Pascal, VB Script, Perl, PHP, Ruby, and/or other programming languages capable of implementing the functionalities described herein.

Where a reference to a specific file or file type is used herein, other files or file types can be substituted.

Where a reference to a data structure is used herein, it should be understood that any variety of data structures can be used such as, for example, array, list, linked list, doubly linked list, queue, tree, heap, graph, map, grid, matrix, multi-dimensional matrix, table, database, database management system (DBMS), file, neural network, and/or any other type or form of a data structure including a custom one. A data structure may include one or more fields or data fields that are part of or associated with the data structure. A field or data field may include a data, an object, a data structure, and/or any other element or a reference/pointer thereto. A data structure can be stored in one or more memories, files, or other repositories. A data structure and/or elements thereof, when stored in a memory, file, or other repository, may be stored in a different arrangement than the arrangement of the data structure and/or elements thereof. For example, a sequence of elements can be stored in an arrangement other

than a sequence in a memory, file, or other repository.

Where a reference to a repository is used herein, it should be understood that a repository may be or include one or more files or file systems, one or more storage locations or structures, one or more storage systems, one or more memory locations or structures, and/or other file, storage, memory, or data arrangements.

Where a reference to an interface is used herein, it should be understood that the interface comprises any hardware, device, system, program, method, and/or combination thereof that enable direct or operative coupling, connection, and/or interaction of the elements between which the interface is indicated. A line or arrow shown in the figures between any of the depicted elements comprises such interface. Examples of an interface include a direct connection, an operative connection, a wired connection (i.e. wire, cable, etc.), a wireless connection, a device, a network, a bus, a circuit, a firmware, a driver, a bridge, a program, a combination thereof, and/or others.

Where a reference to an element coupled or connected to another element is used herein, it should be understood that the element may be in communication or other interactive relationship with the other element. Furthermore, an element coupled or connected to another element can be coupled or connected to any other element in alternate implementations. Terms coupled, connected, interfaced, or other such terms may be used interchangeably herein depending on context.

Where a reference to an element matching another element is used herein, it should be understood that the element may be equivalent or similar to the other element. Therefore, the term match or matching can refer to total equivalence or similarity depending on context.

Where a reference to a device is used herein, it should be understood that the device may include or be referred to as a system, and vice versa depending on context, since a device may include a system of elements and a system may be embodied in a device.

Where a reference to a collection of elements is used herein, it should be understood that the collection of elements may include one or more elements. In some aspects or contexts, a reference to a collection of elements does not imply that the collection is an element itself.

Where a reference to an object is used herein, it should be understood that the object may be a physical object (i.e. object detected in a device's surrounding, etc.), an electronic object (i.e. object in an object oriented application program, etc.), and/or other object depending on context.

Where a mention of a function, method, routine, subroutine, or other such procedure is used herein, it should be understood that the function, method, routine, subroutine, or other such procedure comprises a call, reference, or pointer to the function, method, routine, subroutine, or other such procedure.

Where a mention of data, object, data structure, item, element, or thing is used herein, it should be understood that the data, object, data structure, item, element, or thing comprises a reference or pointer to the data, object, data structure, item, element, or thing.

Referring to Fig. 2, an embodiment of Device 98 comprising Unit for Learning and/or Using a Device's Circumstances for Autonomous Device Operation (DCADO Unit 100) is illustrated. Device 98 also comprises interconnected Processor 11, Human-machine Interface 23, Sensor 92, Object Processing Unit 93, Memory 12, and Storage 27. Processor 11 includes or executes Application Program 18. DCADO Unit 100 comprises interconnected Artificial Intelligence Unit 110, Acquisition Interface 120, and Modification Interface 130. Other additional elements can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized

in alternate embodiments.

In one example, the teaching presented by the disclosure can be implemented in a device or system for learning and/or using a device's circumstances for autonomous device operation. The device or system may include a processor circuit (i.e. Processor 11, etc.) configured to execute instruction sets (i.e. Instruction Sets 526, etc.) for operating a device. The device or system may further include a memory unit (i.e. Memory 12, etc.) configured to store data. The device or system may further include a sensor (i.e. Sensor 92, etc.) configured to detect objects (i.e. Objects 615, etc.). The device or system may further include an artificial intelligence unit (i.e. Artificial Intelligence Unit 110, etc.). The artificial intelligence unit may be configured to receive a first collection of object representations (i.e. Collection of Object Representations 525, etc.), the first collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may also be configured to receive a first one or more instruction sets for operating the device. The artificial intelligence unit may also be configured to learn the first collection of object representations correlated with the first one or more instruction sets for operating the device. The artificial intelligence unit may also be configured to receive a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may also be configured to anticipate the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may also be configured to cause the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations performed in response to the executing by the processor circuit. Any of the operations of the aforementioned elements can be performed repeatedly and/or in different orders in alternate embodiments. In some embodiments, a stream of collections of object representations can be used instead of or in addition to any collection of object representations such as, for example, using a first stream of collections of object representations instead of the first collection of object representations. In other embodiments, a logic circuit (i.e. Logic Circuit 250, etc.) may be used instead of the processor circuit. In such embodiments, one or more instruction sets for operating the device (i.e. first one or more instruction sets for operating the device, etc.) may include or be substituted with one or more inputs into or one or more outputs from the logic circuit. In further embodiments, an actuator may be included instead of or in addition to the processor circuit. In such embodiments, one or more instruction sets for operating the device (i.e. first one or more instruction sets for operating the device, etc.) may include or be substituted with one or more inputs into the actuator. Other additional elements can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate embodiments. The device or system for learning and/or using a device's circumstances for autonomous device operation may include any actions or operations of any of the disclosed methods such as methods 9100, 9200, 9300, 9400, 9500, 9600, and/or others (all later described).

Device 98 comprises any hardware, programs, or a combination thereof. Although, Device 98 is referred to as a device herein, Device 98 may be or include a system as a system may be embodied in Device 98. Device 98 may include any features, functionalities, and embodiments of Computing Device 70, or elements thereof. In some

embodiments, Device 98 includes a computing enabled device for performing mechanical or physical operations (i.e. via actuators, etc.). In other embodiments, Device 98 includes a computing enabled device for performing non-mechanical and/or other operations. Examples of Device 98 include an industrial machine, a toy, a robot, a vehicle, an appliance, a control device, a smartphone or other mobile computer, any computer, and/or other computing enabled device or machine. Such device or machine may be built for any function or purpose some examples of which are described later.

User 50 (also referred to simply as user or other suitable name or reference) comprises a human user or non-human user. A non-human User 50 includes any device, system, program, and/or other mechanism for operating or controlling Device 98 and/or elements thereof. In one example, User 50 may issue an operating direction to Application Program 18 responsive to which Application Program's 18 instructions or instruction sets may be executed by Processor 11 to perform a desired operation on Device 98. In another example, User 50 may issue an operating direction to Processor 11, Logic Circuit 250 (later described), and/or other processing element responsive to which Processor 11, Logic Circuit 250, and/or other processing element may implement logic to perform a desired operation on Device 98. User's 50 operating directions comprise any user inputted data (i.e. values, text, symbols, etc.), directions (i.e. move right, move up, move forward, copy an item, click on a link, etc.), instructions or instruction sets (i.e. manually inputted instructions or instruction sets, etc.), and/or other inputs or information. A non-human User 50 can utilize more suitable interfaces instead of, or in addition to, Human-machine Interface 23 and/or Display 21 for controlling Device 98 and/or elements thereof. Examples of such interfaces include an application programming interface (API), bridge (i.e. bridge between applications, devices, or systems, etc.), driver, socket, direct or operative connection, handle, function/routine/subroutine, and/or other interfaces.

In some embodiments, Processor 11, Logic Circuit 250, Application Program 18, and/or other processing element may control or affect an actuator (not shown). Actuator comprises the functionality for implementing motion, actions, behaviors, maneuvers, and/or other mechanical or physical operations. Device 98 may include one or more actuators to enable Device 98 to perform mechanical, physical, or other operations and/or to interact with its environment. For example, an actuator may include or be coupled to an element such as a wheel, arm, or other element to act upon the environment. Examples of an actuator include a motor, a linear motor, a servomotor, a hydraulic element, a pneumatic element, an electro-magnetic element, a spring element, and/or other actuators. Examples of types of actuators include a rotary actuator, a linear actuator, and/or other types of actuators. In other embodiments, Processor 11, Logic Circuit 250, Application Program 18, and/or other processing element may control or affect any other device or element instead of or in addition to an actuator.

Referring to Figs. 3A-3E, various embodiments of Sensors 92 and elements of Object Processing Unit 93 are illustrated.

Sensor 92 (also referred to simply as sensor or other suitable name or reference) comprises the functionality for obtaining or detecting information about its environment, and/or other functionalities. As such, one or more Sensors 92 can be used to detect objects and/or their properties in Device's 98 surrounding. In some aspects, Device's 98 surrounding may include exterior of Device 98. In other aspects, Device's 98 surrounding may include interior of Device 98 in case of hollow Device 98, Device 98 comprising compartments or openings, and/or other variously shaped Device 98. Examples of aspects of an environment that Sensor 92 can measure or be sensitive to include light (i.e. camera, lidar, etc.), electromagnetism/electromagnetic field (i.e. radar, etc.), sound (i.e.

microphone, sonar, etc.), physical contact (i.e. tactile sensor, etc.), magnetism/magnetic field (i.e. compass, etc.), electricity/electric field, temperature, gravity, vibration, pressure, and/or others. In some aspects, a passive sensor (i.e. camera, microphone, etc.) measures signals or radiation emitted or reflected by an object. In other aspects, an active sensor (i.e. lidar, radar, sonar, etc.) emits signals or radiation and measures the signals or radiation reflected or backscattered from an object. A reference to a Sensor 92 herein includes a reference to one or more Sensors 92 as applicable. In some designs, a plurality of Sensors 92 may be used to detect objects and/or their properties from different angles or sides of Device 98. For example, four Cameras 92a can be placed on four corners of Device 98 to cover 360 degrees of view of Device's 98 surrounding. In other designs, a plurality of different types of Sensors 92 may be used to detect different types of objects and/or their properties. For example, one or more Cameras 92a can be used to detect and identify an object, whereas, Radar 92d can be used to determine distance and bearing/angle of the object relative to Device 98. In further designs, a signal-emitting element can be placed within or onto an object and Sensor 92 can detect the signal from the signal-emitting element, thereby detecting the object and/or its properties. For example, a radio-frequency identification (RFID) emitter may be placed within an object to help Sensor 92 detect, identify, and/or obtain other information about the object.

In some embodiments, Sensor 92 may be or include Camera 92a as shown in Fig. 3A. Camera 92a comprises the functionality for capturing one or more pictures, and/or other functionalities. As such, Camera 92a can be used to capture pictures of Device's 98 surrounding. Camera 92a may be useful in detecting existence of an object, type of an object, identity of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In some aspects, Camera 92a may be or comprises a motion picture camera that can capture streams of pictures (i.e. motion pictures, videos, etc.). In other aspects, Camera 92a may be or comprises a still picture camera that can capture still pictures (i.e. photographs, etc.). In further aspects, Camera 92a may be or comprises a stereo camera (i.e. camera with multiple lenses, etc.) that can capture stereoscopic or range pictures. In further aspects, Camera 92a may be or comprises any other Camera 92a. In general, Camera 92a may capture any light (i.e. visible light, infrared light, ultraviolet light, x-ray light, etc.) across the electromagnetic spectrum onto a light-sensitive material. Any other technique known in art can be utilized to facilitate Camera 92a functionalities. In one example, a digital Camera 92a can utilize a charge coupled device (CCD), a complementary metal–oxide–semiconductor (CMOS) sensor, and/or other electronic image sensor to capture digital pictures that can then be stored in a memory or storage, or transmitted to any of the disclosed or other elements for further processing. In another example, analog Camera 92a can utilize an analog-to-digital converter to produce digital pictures. In some embodiments, Camera 92a can be built, embedded, or integrated in Device 98 and/or other disclosed element. In other embodiments, Camera 92a can be an external Camera 92a connected with Device 98 and/or other disclosed element. In further embodiments, Camera 92a comprises Computing Device 70 or elements thereof. In general, Camera 92a can be implemented in any suitable configuration to provide its functionalities. Camera 92a may capture one or more digital pictures. A digital picture may include a collection of color encoded pixels or dots. Examples of file formats that can be utilized to store a digital picture include JPEG, GIF, TIFF, PNG, PDF, and/or other digitally encoded picture formats. A stream of digital pictures (i.e. motion picture, video, etc.) may include one or more digital pictures. Examples of file formats that can be utilized to store a stream of digital pictures include MPEG, AVI, FLV, MOV, RM, SWF, WMV, DivX, and/or other digitally encoded motion picture formats.

In other embodiments, Sensor 92 may be or include Microphone 92b as shown in Fig. 3B. Microphone 92b comprises the functionality for capturing one or more sounds, and/or other functionalities. As such, Microphone 92b can be used to capture sounds from Device's 98 surrounding. Microphone 92b may be useful in detecting existence of an object, type of an object, identity of an object, bearing/angle of an object, activity (i.e. motion, sounding, etc.) of an object, and/or other properties of an object. In some aspects, Microphone 92b may be omnidirectional microphone that enables capturing sounds from any direction. In other aspects, Microphone 92b may be a directional (i.e. unidirectional, bidirectional, etc.) microphone that enables capturing sounds from one or more directions while ignoring or being insensitive to sounds from other directions. In general, Microphone 92b may utilize a membrane sensitive to air pressure and may produce electrical signal from air pressure variations. Samples of the electrical signal can then be read to produce a stream of digital sound samples. Any other technique known in art can be utilized to facilitate Microphone 92b functionalities. In one example, a digital Microphone 92b may include an integrated analog-to-digital converter to capture a stream of digital sound samples that can then be stored in a memory or storage, or transmitted to any of the disclosed or other elements for further processing. In another example, analog Microphone 92b may utilize an external analog-to-digital converter to produce a stream of digital sound samples. In some embodiments, Microphone 92b can be built, embedded, or integrated in Device 98. In other embodiments, Microphone 92b can be an external Microphone 92b connected with Device 98. In further embodiments where used in water, Microphone 92b may be or include a hydrophone. In further embodiments, Microphone 92b comprises Computing Device 70 or elements thereof. In general, Microphone 92b can be implemented in any suitable configuration to provide its functionalities. Examples of file formats that can be utilized to store a stream of digital sound samples include WAV, WMA, AIFF, MP3, RA, OGG, and/or other digitally encoded sound formats.

In further embodiments, Sensor 92 may be or include Lidar 92c as shown in Fig. 3C. Lidar 92c may be useful in detecting existence of an object, type of an object, identity of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In some aspects, Lidar 92c may emit a light signal (i.e. laser beam, etc.) and listen for a signal that is reflected or backscattered from an object. Any other technique known in art can be utilized to facilitate Lidar 92c functionalities.

In further embodiments, Sensor 92 may be or include a Radar 92d as shown in Fig. 3D. Radar 92d may be useful in detecting existence of an object, type of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In some aspects, Radar 92d may emit a radio signal (i.e. radio wave, etc.) and listen for a signal that is reflected or backscattered from an object. Any other technique known in art can be utilized to facilitate Radar 92d functionalities.

In further embodiments, Sensor 92 may be or include Sonar 92e as shown in Fig. 3E. Sonar 92e may be useful in detecting existence of an object, type of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In some aspects, Sonar 92e may emit a sound signal (i.e. sound pulse, etc.) and listen for a signal that is reflected or backscattered from an object. Any other technique known in art can be utilized to facilitate Sonar 92e functionalities.

One of ordinary skill in art will understand that the aforementioned sensors are described merely as examples of a variety of possible implementations, and that while all possible sensors are too voluminous to describe, other sensors known in art that can facilitate detecting of objects and/or their properties in Device's 98

surrounding are within the scope of this disclosure. Any combination of the aforementioned and/or other sensors can be used in various embodiments.

Object Processing Unit 93 comprises the functionality for processing output from Sensor 92 to obtain information of interest, and/or other functionalities. As such, Object Processing Unit 93 can be used to process output from Sensor 92 to detect objects and/or their properties in Device's 98 surrounding. In some embodiments, Object Processing Unit 93 comprises the functionality for creating or generating Collection of Object Representations 525 (also referred to as Coll of Obj Rep or other suitable name or reference) and storing one or more Object Representations 625 (also referred to simply as object representations, representations of objects, or other suitable name or reference), Object Properties 630 (also referred to simply as object properties or other suitable name or reference), and/or other elements or information into the Collection of Object Representations 525. As such, Collection of Object Representations 525 comprises the functionality for storing one or more Object Representations 625, Object Properties 630, and/or other elements or information. Object Representation 625 may include an electronic representation of an object (i.e. Object 615 [later described], etc.) detected in Device's 98 surrounding. In some aspects, Collection of Object Representations 525 includes one or more Object Representations 625, Object Properties 630, and/or other elements or information related to objects detected in Device's 98 surrounding at a particular time. Collection of Object Representations 525 may, therefore, include knowledge (i.e. unit of knowledge, etc.) of Device's 98 circumstances including objects with various properties at a particular time. In some designs, a Collection of Object Representations 525 may include or be associated with a time stamp (not shown), order (not shown), or other time related information. For example, one Collection of Object Representations 525 may be associated with time stamp t1, another Collection of Object Representations 525 may be associated with time stamp t2, and so on. Time stamps t1, t2, etc. may indicate the times of generating Collections of Object Representations 525, for instance. In other embodiments, Object Processing Unit 93 comprises the functionality for creating or generating a stream of Collections of Object Representations 525. A stream of Collections of Object Representations 525 may include one Collection of Object Representations 525 or a group, sequence, or other plurality of Collections of Object Representations 525. In some aspects, a stream of Collections of Object Representations 525 includes one or more Collections of Object Representations 525, and/or other elements or information related to objects detected in Device's 98 surrounding over time. A stream of Collections of Object Representations 525 may, therefore, include knowledge (i.e. unit of knowledge, etc.) of Device's 98 circumstances including objects with various properties over time. As circumstances including objects with various properties in Device's 98 surrounding change (i.e. objects and/or their properties change, move, act, transform, etc.) over time, this change may be captured in a stream of Collections of Object Representations 525. In some designs, each Collection of Object Representations 525 in a stream may include or be associated with the aforementioned time stamp, order, or other time related information. For example, one Collection of Object Representations 525 in a stream may be associated with order 1, a next Collection of Object Representations 525 in the stream may be associated with order 2, and so on. Orders 1, 2, etc. may indicate the orders or places of Collections of Object Representations 525 within a stream (i.e. sequence, etc.), for instance. Examples of objects include biological objects (i.e. persons, animals, vegetation, etc.), nature objects (i.e. rocks, bodies of water, etc.), manmade objects (i.e. buildings, streets, ground/aerial/aquatic vehicles, etc.), and/or others. In some aspects, any part of an object may be detected as an object itself. For instance, instead of or in addition to detecting a vehicle as

an object, a wheel and/or other parts of the vehicle may be detected as objects. In general, object may include any object or part thereof that can be detected. Examples of object properties include existence of an object, type of an object (i.e. person, cat, vehicle, building, street, tree, rock, etc.), identity of an object (i.e. name, identifier, etc.), distance of an object, bearing/angle of an object, location of an object (i.e. distance and bearing/angle from a known point, coordinates, etc.), shape/size of an object (i.e. height, width, depth, computer model, point cloud, etc.), activity of an object (i.e. motion, gestures, etc.), and/or other properties of an object. Type of an object, for example, may include any classification of objects ranging from detailed such as person, cat, vehicle, building, street, tree, rock, etc. to generalized such as biological object, nature object, manmade object, etc., and/or others including their sub-types. Location of an object, for example, can include a relative location such as one defined by distance and bearing/angle from a known point or location (i.e. Device 98 location, etc.). Location of an object, for example, can also include absolute location such as one defined by object coordinates. In general, an object property may include any attribute of an object (i.e. existence of an object, type of an object, identity of an object, shape/size of an object, etc.), any relationship of an object with Device 98, other objects, or the environment (i.e. distance of an object, bearing/angle of an object, friend/foe relationship, etc.), and/or other information related to an object. In some implementations, Object Processing Unit 93 and/or any of its elements or functionalities can be included in Sensor 92. In other implementations, Object Processing Unit 93 and/or any of its elements or functionalities can be embedded into or operate on Processor 11. In further implementations, Object Processing Unit 93 and/or any of its elements or functionalities can be embedded into or operate in DCADO Unit 100, and/or other disclosed elements. Object Processing Unit 93 may be provided in any suitable configuration. Object Processing Unit 93 may include any signal processing techniques or elements known in art as applicable.

In some embodiments, Object Processing Unit 93 may include Picture Recognizer 94a as shown in Fig. 3A. Picture Recognizer 94a comprises the functionality for detecting or recognizing objects and/or their properties in visual data, and/or other disclosed functionalities. Visual data includes digital motion pictures, digital still pictures, and/or other visual data. Examples of file formats that can be utilized to store visual data include AVI, DivX, MPEG, JPEG, GIF, TIFF, PNG, PDF, and/or other file formats. For example, Picture Recognizer 94a can be used for detecting or recognizing objects and/or their properties in one or more digital pictures captured by one or more Cameras 92a. Picture Recognizer 94a can be utilized in detecting or recognizing existence of an object, type of an object, identity of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In general, Picture Recognizer 94a can be used for any operation supported by Picture Recognizer 94a. Picture Recognizer 94a may detect or recognize an object and/or its properties as well as track the object and/or its properties in one or more digital pictures or streams of digital pictures (i.e. motion pictures, video, etc.). In the case of a person, Picture Recognizer 94a may detect or recognize a human head or face, upper body, full body, or portions/combinations thereof. In some aspects, Picture Recognizer 94a may detect or recognize objects and/or their properties from a digital picture by comparing regions of pixels from the digital picture with collections of pixels comprising known objects and/or their properties. The collections of pixels comprising known objects and/or their properties can be learned or manually, programmatically, or otherwise defined. The collections of pixels comprising known objects and/or their properties can be stored in any data structure or repository (i.e. one or more files, database, etc.) that resides locally on Device 98, or remotely on a remote computing device (i.e. server, cloud, etc.) accessible over a network or an interface. In other aspects, Picture

Recognizer 94a may detect or recognize objects and/or their properties from a digital picture by comparing features (i.e. lines, edges, ridges, corners, blobs, regions, etc.) of the digital picture with features of known objects and/or their properties. The features of known objects and/or their properties can be learned or manually, programmatically, or otherwise defined. The features of known objects and/or their properties can be stored in any data structure or repository (i.e. neural network, one or more files, database, etc.) that resides locally on Device 98, or remotely on a remote computing device (i.e. server, cloud, etc.) accessible over a network or an interface. Typical steps or elements in a feature oriented picture recognition include pre-processing, feature extraction, detection/segmentation, decision-making, and/or others, or a combination thereof, each of which may include its own sub-steps or sub-elements depending on the application. In further aspects, Picture Recognizer 94a may detect or recognize multiple objects and/or their properties from a digital picture using the aforementioned pixel or feature comparisons, and/or other detection or recognition techniques. For example, a picture may depict two objects in two of its regions both of which Picture Recognizer 94a can detect simultaneously. In further aspects, where objects and/or their properties span multiple pictures, Picture Recognizer 94a may detect or recognize objects and/or their properties by applying the aforementioned pixel or feature comparisons and/or other detection or recognition techniques over a stream of digital pictures (i.e. motion picture, video, etc.). For example, once an object is detected in a digital picture (i.e. frame, etc.) of a stream of digital pictures (i.e. motion picture, video, etc.), the region of pixels comprising the detected object or the object's features can be searched in other pictures of the stream of digital pictures, thereby tracking the object through the stream of digital pictures. In further aspects, Picture Recognizer 94a may detect or recognize an object's activities by identifying and/or analyzing differences between a detected region of pixels of one picture (i.e. frame, etc.) and detected regions of pixels of other pictures in a stream of digital pictures. For example, a region of pixels comprising a person's face can be detected in multiple consecutive pictures of a stream of digital pictures (i.e. motion picture, video, etc.). Differences among the detected regions of the consecutive pictures may be identified in the mouth part of the person's face to indicate smiling or speaking activity. In further aspects, Picture Recognizer 94a may detect or recognize objects and/or their properties using one or more artificial neural networks, which may include statistical techniques. Examples of artificial neural networks that can be used in Picture Recognizer 94a include convolutional neural networks (CNNs), time delay neural networks (TDNNs), deep neural networks, and/or others. In one example, picture recognition techniques and/or tools involving convolutional neural networks may include identifying and/or analyzing tiled and/or overlapping regions or features of a digital picture, which may then be used to search for pictures with matching regions or features. In another example, features of different convolutional neural networks responsible for spatial and temporal streams can be fused to detect objects and/or their properties in streams of digital pictures (i.e. motion pictures, videos, etc.). In general, Picture Recognizer 94a may include any machine learning, deep learning, and/or other artificial intelligence techniques. In further aspects, Picture Recognizer 94a can detect distance of a recognized object in a picture captured by a camera using structured light, sheet of light, or other lighting schemes, and/or by using phase shift analysis, time of flight, interferometry, or other techniques. In further aspects, Picture Recognizer 94a may detect distance of a recognized object in a picture captured by a stereo camera by using triangulation and/or other techniques. In further aspects, Picture Recognizer 94a may detect bearing/angle of a recognized object relative to the camera-facing direction by measuring the distance from the vertical centerline of the picture to a pixel in the recognized object based on known picture resolution and camera's angle of view. Any other techniques known in art can be utilized in Picture

Recognizer 94a. For example, thresholds for similarity, statistical techniques, and/or optimization techniques can be utilized to determine a match in any of the above-described detection or recognition techniques. In some exemplary embodiments, object recognition techniques and/or tools such as OpenCV (Open Source Computer Vision) library, CamFind API, Kooaba, 6px API, Dextro API, and/or others can be utilized for detecting or recognizing objects and/or their properties in digital pictures. In some aspects, picture recognition techniques and/or tools involve identifying and/or analyzing features such as lines, edges, ridges, corners, blobs, regions, and/or their relative positions, sizes, shapes, etc., which may then be used to search for pictures with matching features. For example, OpenCV library can detect an object (i.e. person, animal, vehicle, rock, etc.) and/or its properties in one or more digital pictures captured by Camera 92a or stored in an electronic repository, which can then be utilized in DCADO Unit 100, Artificial Intelligence Unit 110, and/or other elements. In other exemplary embodiments, facial recognition techniques and/or tools such as OpenCV (Open Source Computer Vision) library, Animetrics FaceR API, Lambda Labs Facial Recognition API, Face++ SDK, Neven Vision (also known as N-Vision) Engine, and/or others can be utilized for detecting or recognizing faces in digital pictures. In some aspects, facial recognition techniques and/or tools involve identifying and/or analyzing facial features such as the relative position, size, and/or shape of the eyes, nose, cheekbones, jaw, etc., which may then be used to search for pictures with matching features. For example, FaceR API can detect a person's face in one or more digital pictures captured by Camera 92a or stored in an electronic repository, which can then be utilized in DCADO Unit 100, Artificial Intelligence Unit 110, and/or other elements.

Various aspects or properties of digital pictures or pixels can be taken into account by Picture Recognizer 94a in any of the recognizing or comparisons. Examples of such aspects or properties include color adjustment, size adjustment, content manipulation, transparency (i.e. alpha channel, etc.), use of mask, and/or others. In some implementations, as digital pictures can be captured by various picture taking equipment, in various environments, and under various lighting conditions, Picture Recognizer 94a can adjust lighting or color of pixels or otherwise manipulate pixels before or during comparison. Lighting or color adjustment (also referred to as gray balance, neutral balance, white balance, etc.) may generally include manipulating or rebalancing the intensities of the colors (i.e. red, green, and/or blue if RGB color model is used, etc.) of one or more pixels. For example, Picture Recognizer 94a can adjust lighting or color of some or all pixels of one picture to make it more comparable to another picture. Picture Recognizer 94a can also incrementally adjust the pixels such as increasing or decreasing the red, green, and/or blue pixel values by a certain amount in each cycle of comparisons in order to find a substantially similar match at one of the incremental adjustment levels. Any of the publically available, custom, or other lighting or color adjustment techniques or programs can be utilized such as color filters, color balancing, color correction, and/or others. In other implementations, Picture Recognizer 94a can resize or otherwise transform a digital picture before or during comparison. Such resizing or transformation may include increasing or decreasing the number of pixels of a digital picture. For example, Picture Recognizer 94a can increase or decrease the size of a digital picture proportionally (i.e. increase or decrease length and/or width keeping aspect ratio constant, etc.) to equate its size with the size of another digital picture. Picture Recognizer 94a can also incrementally resize a digital picture such as increasing or decreasing the size of the digital picture proportionally by a certain amount in each cycle of comparisons in order to find a substantially similar match at one of the incremental sizes. Any of the publically available, custom, or other digital picture resizing techniques or programs can be utilized such as nearest-neighbor

interpolation, bilinear interpolation, bicubic interpolation, and/or others. In further implementations, Picture Recognizer 94a can manipulate content (i.e. all pixels, one or more regions, one or more depicted objects, etc.) of a digital picture before or during comparison. Such content manipulation may include moving, centering, aligning, resizing, transforming, and/or otherwise manipulating content of a digital picture. For example, Picture Recognizer 94a can move, center, or align content of one picture to make it more comparable to another picture. Any of the publically available, custom, or other digital picture manipulation techniques or programs can be utilized such as pixel moving, warping, distorting, aforementioned interpolations, and/or others. In further implementations, in digital pictures comprising transparency features or functionalities, Picture Recognizer 94a can utilize a threshold for acceptable number or percentage transparency difference. Alternatively, transparency can be applied to one or more pixels of a digital picture and color difference may then be determined between compared pixels taking into account the transparency related color effect. Alternatively, transparent pixels can be excluded from comparison. In further implementations, certain regions or subsets of pixels can be ignored or excluded during comparison using a mask. In general, any region or subset of a picture determined to contain no content of interest can be excluded from comparison using a mask. Examples of such regions or subsets include background, transparent or partially transparent regions, regions comprising insignificant content, or any arbitrary region or subset. Picture Recognizer 94a can perform any other pre-processing or manipulation of digital pictures or pixels before or during recognizing or comparison.

In other embodiments, Object Processing Unit 93 may include Sound Recognizer 94b as shown in Fig. 3B. Sound Recognizer 94b comprises the functionality for detecting or recognizing objects and/or their properties in audio data, and/or other disclosed functionalities. Audio data includes digital sound, and/or other audio data. Examples of file formats that can be utilized to store audio data include WAV, WMA, AIFF, MP3, RA, OGG, and/or other file formats. For example, Sound Recognizer 94b can be used for detecting or recognizing objects and/or their properties in a stream of digital sound samples captured by one or more Microphones 92b. In the case of a person, Sound Recognizer 94b may detect or recognize human voice. Sound Recognizer 94b can be utilized in detecting or recognizing existence of an object, type of an object, identity of an object, bearing/angle of an object, activity (i.e. motion, sounding, etc.) of an object, and/or other properties of an object. In general, Sound Recognizer 94b can be used for any operation supported by Sound Recognizer 94b. In some aspects, Sound Recognizer 94b may detect or recognize an object and/or its properties from a stream of digital sound samples by comparing collections of sound samples from the stream of digital sound samples with collections of sound samples of known objects and/or their properties. The collections of sound samples of known objects and/or their properties can be learned, or manually, programmatically, or otherwise defined. The collections of sound samples of known objects and/or their properties can be stored in any data structure or repository (i.e. one or more files, database, etc.) that resides locally on Device 98, or remotely on a remote computing device (i.e. server, cloud, etc.) accessible over a network or an interface. In other aspects, Sound Recognizer 94b may detect or recognize an object and/or its properties from a stream of digital sound samples by comparing features from the stream of digital sound samples with features of sounds of known objects and/or their properties. The features of sounds of known objects and/or their properties can be learned, or manually, programmatically, or otherwise defined. The features of sounds of known objects and/or their properties can be stored in any data structure or repository (i.e. one or more files, database, neural network, etc.) that resides locally on Device 98, or remotely on a remote computing device (i.e. server, cloud, etc.) accessible over

a network or an interface. Typical steps or elements in a feature oriented sound recognition include pre-processing, feature extraction, acoustic modeling, language modeling, and/or others, or a combination thereof, each of which may include its own sub-steps or sub-elements depending on the application. In further aspects, Sound Recognizer 94b may detect or recognize a variety of sounds from a stream of digital sound samples using the aforementioned sound sample or feature comparisons, and/or other detection or recognition techniques. For example, sound of a person, animal, vehicle, and/or other sounds can be detected by Sound Recognizer 94b. In further aspects, Sound Recognizer 94b may detect or recognize sounds using Hidden Markov Models (HMM), Artificial Neural Networks, Dynamic Time Warping (DTW), Gaussian Mixture Models (GMM), and/or other models or techniques, or a combination thereof. Some or all of these models or techniques may include statistical techniques. Examples of artificial neural networks that can be used in Sound Recognizer 94b include recurrent neural networks, time delay neural networks (TDNNs), deep neural networks, convolutional neural networks, and/or others. In general, Sound Recognizer 94b may include any machine learning, deep learning, and/or other artificial intelligence techniques. In further aspects, Sound Recognizer 94b may detect bearing/angle of a recognized object by measuring the direction in which Microphone 92b is pointing when sound of maximum strength is received, by analyzing amplitude of the sound, by performing phase analysis (i.e. with microphone array, etc.) of the sound, and/or by utilizing other techniques. Any other techniques known in art can be utilized in Sound Recognizer 94b. For example, thresholds for similarity, statistical techniques, and/or optimization techniques can be utilized to determine a match in any of the above-described detection or recognition techniques. In some exemplary embodiments, operating system's Sound recognition functionalities such as iOS's Voice Services, Siri, and/or others can be utilized in Sound Recognizer 94b. For example, iOS Voice Services can detect an object (i.e. person, etc.) and/or its properties in a stream of digital sound samples captured by Microphone 92b or stored in an electronic repository, which can then be utilized in DCADO Unit 100, Artificial Intelligence Unit 110, and/or other elements. In other exemplary embodiments, Java Speech API (JSAPI) implementation such as The Cloud Garden, Sphinx, and/or others can be utilized in Sound Recognizer 94b. For example, Cloud Garden JSAPI can detect an object (i.e. person, animal, vehicle, etc.) and/or its properties in a stream of digital sound samples captured by Microphone 92b or stored in an electronic repository, which can then be utilized in DCADO Unit 100, Artificial Intelligence Unit 110, and/or other elements. Any other programming language's or platform's speech or sound processing API can similarly be utilized. In further exemplary embodiments, applications or engines providing Sound recognition functionalities such as HTK (Hidden Markov Model Toolkit), Kaldi, OpenEars, Dragon Mobile, Julius, iSpeech, CeedVocal, and/or others can be utilized in Sound Recognizer 94b. For example, Kaldi SDK can detect an object (i.e. person, animal, vehicle, etc.) and/or its properties in a stream of digital sound samples captured by Microphone 92b or stored in an electronic repository, which can then be utilized in DCADO Unit 100, Artificial Intelligence Unit 110, and/or other elements.

In further embodiments, Object Processing Unit 93 may include Lidar Processing Unit 94c as shown in Fig. 3C. Lidar Processing Unit 94c comprises the functionality for detecting or recognizing objects and/or their properties using light, and/or other disclosed functionalities. As such, Lidar Processing Unit 94c can be utilized in detecting existence of an object, type of an object, identity of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In general, Lidar Processing Unit 94c can be used for any operation supported by Lidar Processing Unit 94c. In one example, Lidar Processing Unit 94c may detect distance of an object by measuring time delay between emission of a light

signal (i.e. laser beam, etc.) and return of the light signal reflected from the object based on known speed of light. In another example, Lidar Processing Unit 94c may detect bearing/angle of an object by analyzing the amplitudes of a light signal received by an array of detectors (i.e. detectors arranged into a quadrant or other arrangement, etc.). In a further example, Lidar Processing Unit 94c may detect existence, type, identity, shape/size, activity, and/or other properties of an object by illuminating the object with light and acquiring an image of the object, which can then be processed using some of the previously described or other picture recognition techniques. In a further example, Lidar Processing Unit 94c may detect existence, type, identity, shape/size, activity, and/or other properties of an object by illuminating the object with light and acquiring a point cloud representation of the object. Lidar Processing Unit 94c may detect objects and/or their properties by utilizing any lidar or light-related techniques known in art.

In further embodiments, Object Processing Unit 93 may include Radar Processing Unit 94d as shown in Fig. 3D. Radar Processing Unit 94d comprises the functionality for detecting or recognizing objects and/or their properties using radio waves, and/or other disclosed functionalities. As such, Radar Processing Unit 94d can be utilized in detecting existence of an object, type of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In general, Radar Processing Unit 94d can be used for any operation supported by Radar Processing Unit 94d. In one example, Radar Processing Unit 94d may detect existence of an object by emitting a radio signal and listening for the radio signal reflected from the object. In another example, Radar Processing Unit 94d may detect distance of an object by measuring time delay between emission of a radio signal and return of the radio signal reflected from the object based on known speed of the radio signal. In a further example, Radar Processing Unit 94d may detect bearing/angle of an object by measuring the direction in which the antenna is pointing when the return signal of maximum strength is received, by analyzing amplitude of the return signal, by performing phase analysis (i.e. with antenna array, etc.) of the return signal, and/or by utilizing any amplitude, phase, or other techniques. In a further example, Radar Processing Unit 94d may detect existence, type, identity, shape/size, activity, and/or other properties of an object by illuminating the object with radio waves and acquiring an image of the object, which can then be processed using some of the previously described or other picture recognition techniques. Radar Processing Unit 94d may detect objects and/or their properties by utilizing any radar or radio-related techniques known in art.

In further embodiments, Object Processing Unit 93 may include Sonar Processing Unit 94e as shown in Fig. 3E. Sonar Processing Unit 94e comprises the functionality for detecting or recognizing objects and/or their properties using sound, and/or other disclosed functionalities. As such, Sonar Processing Unit 94e can be utilized in detecting existence of an object, type of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In general, Sonar Processing Unit 94e can be used for any operation supported by Sonar Processing Unit 94e. In one example, Sonar Processing Unit 94e may detect existence of an object by emitting a sound signal and listening for the sound signal reflected from the object. In another example, Sonar Processing Unit 94e may detect distance of an object by measuring time delay between emission of a sound signal and return of the sound signal reflected from the object based on known speed of the sound signal. In a further example, Sonar Processing Unit 94e may detect bearing/angle of an object by measuring the direction in which the microphone is pointing when the return signal of maximum strength is received, by analyzing amplitude of the return signal, by performing phase analysis (i.e. with

microphone array, etc.) of the return signal, and/or by utilizing any amplitude, phase, or other techniques. In a further example, Sonar Processing Unit 94e may detect existence, type, identity, shape/size, activity, and/or other properties of an object by illuminating the object with sound pulses and acquiring an image of the object, which can then be processed using some of the previously described or other picture recognition techniques. Sonar Processing Unit 94e may detect objects and/or their properties by utilizing any sonar or sound-related techniques known in art.

One of ordinary skill in art will understand that the aforementioned techniques for detecting or recognizing objects and/or their properties are described merely as examples of a variety of possible implementations, and that while all possible techniques for detecting or recognizing objects and/or their properties are too voluminous to describe, other techniques for detecting or recognizing objects and/or their properties known in art are within the scope of this disclosure. Also, any signal processing technique known in art that can facilitate the disclosed functionalities can be utilized in various embodiments. Any combination of the aforementioned and/or other sensors, object detecting or recognizing techniques, signal processing techniques, and/or other elements or techniques can be used in various embodiments.

Referring to Figs. 4A-4B, an exemplary embodiment of Objects 615 (also referred to simply as objects or other suitable name or reference) detected in Device's 98 surrounding, and resulting Collection of Object Representations 525 are illustrated.

As shown for example in Fig. 4A, Object 615a is detected. Object 615a may be recognized as a cat. Object 615a may be detected at a distance of 6m from Device 98. Object 615a may be detected at a bearing/angle of 56° from Device's 98 centerline. Furthermore, Object 615b is also detected. Object 615b may be recognized as a tree. Object 615b may be detected at a distance of 10m from Device 98. Object 615b may be detected at a bearing/angle of 131° from Device's 98 centerline. Furthermore, Object 615c is also detected. Object 615c may be recognized as a person. Object 615c may be identified as John Doe. Object 615c may be detected at a distance of 8m from Device 98. Object 615c may be detected at a bearing/angle of 287° from Device's 98 centerline. Any other Objects 615 instead of or in addition to Object 615a, Object 615b, and Object 615c may be detected. In some aspects, any features, functionalities, and embodiments of Camera 92a/Picture Recognizer 94a, Microphone 92b/Sound Recognizer 94b, and/or other sensors or techniques can be utilized for recognizing and/or identifying a person, a cat, a tree, and/or other Objects 615. In further aspects, any features, functionalities, and embodiments of Camera 92a/Picture Recognizer 94a, Microphone 92b/Sound Recognizer 94b, Lidar 92c/Lidar Processing Unit 94c, Radar 92d/Radar Processing Unit 94d, Sonar 92e/Sonar Processing Unit 94e, and/or other sensors or techniques can be utilized for detecting distance, bearing/angle, and/or other object properties.

As shown for example in Fig. 4B, Object Processing Unit 93 may create or generate Collection of Object Representations 525 including Object Representation 625a representing Object 615a, Object Representation 625b representing Object 615b, Object Representation 625c representing Object 615c, etc. For instance, Object Representation 625a may include Object Property 630aa "Cat" in Category 635aa "Type", Object Property 630ab "6m" in Category 635ab "Distance", Object Property 630ac "56°" in Category 635ac "Bearing", etc. Also, Object Representation 625b may include Object Property 630ba "Tree" in Category 635ba "Type", Object Property 630bb "10m" in Category 635bb "Distance", Object Property 630bc "131°" in Category 635bc "Bearing", etc. Also, Object Representation 625c may include Object Property 630ca "Person" in Category 635ca "Type", Object Property 630cb "John Doe" in Category 635cb "Identity", Object Property 630cc "8m" in Category 635cc "Distance", Object Property

630cd "287°" in Category 635cd "Bearing", etc. Any number of Object Representations 625, and/or other elements or information can be included in Collection of Object Representations 525. Any number of Object Properties 630 (also referred to simply as object properties or other suitable name or reference), and/or other elements or information can be included in an Object Representation 625. In some aspects, a reference to Collection of Object Representations 525 comprises a reference to a collection of Object Properties 630 and/or other elements or information related to one or more Objects 615. Other additional Object Representations 625, Object Properties 630, elements, and/or information can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate embodiments of Collection of Object Representations 525.

Referring now to DCADO Unit 100, DCADO Unit 100 comprises any hardware, programs, or a combination thereof. DCADO Unit 100 comprises the functionality for learning the operation of Device 98 in circumstances including objects with various properties. DCADO Unit 100 comprises the functionality for structuring and/or storing this knowledge in a knowledgebase (i.e. neural network, graph, sequences, other repository, etc.). DCADO Unit 100 comprises the functionality for enabling autonomous operation of Device 98 in circumstances including objects with various properties. DCADO Unit 100 comprises the functionality for interfacing with or attaching to Application Program 18, Processor 11, Logic Circuit 250 (later described), and/or other processing element. DCADO Unit 100 comprises the functionality for obtaining instruction sets, data, and/or other information used, implemented, and/or executed by Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. DCADO Unit 100 comprises the functionality for modifying instruction sets, data, and/or other information used, implemented, and/or executed by Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. DCADO Unit 100 comprises learning, anticipating, decision making, automation, and/or other functionalities disclosed herein. Statistical, artificial intelligence, machine learning, and/or other models or techniques are utilized to implement the disclosed devices, systems, and methods.

When DCADO Unit 100 functionalities are applied on Application Program 18, Processor 11, Logic Circuit 250 (later described), and/or other processing element of Device 98, Device 98 may become autonomous. DCADO Unit 100 may take control from, share control with, and/or release control to Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element to implement autonomous operation of Device 98. DCADO Unit 100 may take control from, share control with, and/or release control to Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element automatically or after prompting User 50 to allow it. In some aspects, Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element of an autonomous Device 98 may include or be provided with anticipatory (also referred to as alternate or other suitable name or reference) instructions or instruction sets that User 50 did not issue or cause to be executed. Such anticipatory instructions or instruction sets include instruction sets that User 50 may want or is likely to issue or cause to be executed. Anticipatory instructions or instruction sets can be generated by DCADO Unit 100 or elements thereof based on Device's 98 circumstances including objects with various properties. As such, Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element of an autonomous Device 98 may include or be provided with some or all original instructions or instruction sets and/or any anticipatory instructions or instruction sets generated by DCADO Unit 100. Therefore, autonomous Device 98 operating may include executing some or all original instructions or instruction sets and/or any anticipatory instructions or instruction sets generated by DCADO Unit 100. In one example, DCADO Unit 100 can overwrite or rewrite the original instructions or instruction sets of Application

Program 18, Processor 11, Logic Circuit 250, and/or other processing element with DCADO Unit 100-generated instructions or instruction sets. In another example, DCADO Unit 100 can insert or embed DCADO Unit 100-generated instructions or instruction sets among the original instructions or instruction sets of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. In a further example, DCADO Unit 100 can branch, redirect, or jump to DCADO Unit 100-generated instructions or instruction sets from the original instructions or instruction sets of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element.

In some embodiments, autonomous Device 98 operating comprises determining, by DCADO Unit 100, a next instruction or instruction set to be executed based on Device's 98 circumstances including objects with various properties prior to the user issuing or causing to be executed the next instruction or instruction set. In yet other embodiments, autonomous application operating comprises determining, by DCADO Unit 100, a next instruction or instruction set to be executed based on Device's 98 circumstances including objects with various properties prior to the system receiving the next instruction or instruction set.

In some embodiments, autonomous Device 98 operating includes a partially or fully autonomous operating. In an example involving partially autonomous Device 98 operating, a user confirms DCADO Unit 100-generated instructions or instruction sets prior to their execution. In an example involving fully autonomous application operating, DCADO Unit 100-generated instructions or instruction sets are executed without user or other system confirmation (i.e. automatically, etc.).

In some embodiments, a combination of DCADO Unit 100 and other systems and/or techniques can be utilized to implement Device's 98 operation. In one example, DCADO Unit 100 may be a primary or preferred system for implementing Device's 98 operation. While operating autonomously under the control of DCADO Unit 100, Device 98 may encounter a circumstance including objects with various properties that has not been encountered or learned before. In such situations, User 50 and/or non-DCADO system may take control of Device's 98 operation. DCADO Unit 100 may take control again when Device 98 encounters a previously learned circumstance including objects with various properties. Naturally, DCADO Unit 100 can learn Device's 98 operation in circumstances while User 50 and/or non-DCADO system is in control of Device 98, thereby reducing or eliminating the need for future involvement of User 50 and/or non-DCADO system. In another example, User 50 and/or non-DCADO system may be a primary or preferred system for implementing Device's 98 operation. While operating under the control of User 50 and/or non-DCADO system, User 50 and/or non-DCADO system may release control to DCADO Unit 100 for any reason (i.e. User 50 gets tired or distracted, non-DCADO system gets stuck or cannot make a decision, etc.), at which point Device 98 can be controlled by DCADO Unit 100. In some designs, DCADO Unit 100 may take control in certain special circumstances including objects with various properties where DCADO Unit 100 may offer superior performance even though User 50 and/or non-DCADO system may generally be preferred. Once Device 98 leaves such special circumstances, DCADO Unit 100 may release control to User 50 and/or non-DCADO system. In general, DCADO Unit 100 can take control from, share control with, or release control to User 50, non-DCADO system, and/or other system or process at any time, in any circumstances, and remain in control for any period of time as needed.

In some embodiments, DCADO Unit 100 may control one or more sub-devices, sub-systems, or elements of Device 98 while User 50 and/or non-DCADO system may control other one or more sub-devices, sub-systems, or elements of Device 98.

It should be understood that a reference to autonomous operating of Device 98 may include autonomous operating of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element depending on context.

Referring now to Acquisition Interface 120, Acquisition Interface 120 comprises the functionality for obtaining and/or receiving instruction sets, data, and/or other information. Acquisition Interface 120 comprises the functionality for obtaining and/or receiving instruction sets, data, and/or other information from Processor 11, Application Program 18, Logic Circuit 250 (later described), and/or other processing element. Acquisition Interface 120 comprises the functionality for obtaining and/or receiving instruction sets, data, and/or other information at runtime. In some aspects, an instruction set may include any computer command, instruction, signal, or input used in Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. Therefore, the terms instruction set, command, instruction, signal, input, or other such terms may be used interchangeably herein depending on context. Acquisition Interface 120 also comprises the functionality for attaching to or interfacing with Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. In one example, Acquisition Interface 120 comprises the functionality to access and/or read runtime engine/environment, virtual machine, operating system, compiler, just-in-time (JIT) compiler, interpreter, translator, execution stack, file, object, data structure, and/or other computing system elements. In another example, Acquisition Interface 120 comprises the functionality to access and/or read memory, storage, bus, interfaces, and/or other computing system elements. In a further example, Acquisition Interface 120 comprises the functionality to access and/or read Processor 11 registers and/or other Processor 11 elements. In a further example, Acquisition Interface 120 comprises the functionality to access and/or read inputs and/or outputs of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. In a further example, Acquisition Interface 120 comprises the functionality to access and/or read functions, methods, procedures, routines, subroutines, and/or other elements of Application Program 18. In a further example, Acquisition Interface 120 comprises the functionality to access and/or read source code, bytecode, compiled, interpreted, or otherwise translated code, machine code, and/or other code. In a further example, Acquisition Interface 120 comprises the functionality to access and/or read values, variables, parameters, and/or other data or information. Acquisition Interface 120 also comprises the functionality for transmitting the obtained instruction sets, data, and/or other information to Artificial Intelligence Unit 110 and/or other element. As such, Acquisition Interface 120 provides input into Artificial Intelligence Unit 110 for knowledge structuring, anticipating, decision making, and/or other functionalities later in the process. Acquisition Interface 120 also comprises other disclosed functionalities.

Acquisition Interface 120 can employ various techniques for obtaining instruction sets, data, and/or other information. In one example, Acquisition Interface 120 can attach to and/or obtain Processor's 11, Application Program's 18, Logic Circuit's 250, and/or other processing element's instruction sets, data, and/or other information through tracing or profiling techniques. Tracing or profiling may be used for outputting Processor's 11, Application Program's 18, Logic Circuit's 250, and/or other processing element's instruction sets, data, and/or other information at runtime. For instance, tracing or profiling may include adding trace code (i.e. instrumentation, etc.) to an application and/or outputting trace information to a specific target. The outputted trace information (i.e. instruction sets, data, and/or other information, etc.) can then be provided to or recorded into a file, data structure, repository, an application, and/or other system or target that may receive such trace information. As such, Acquisition Interface

120 can utilize tracing or profiling to obtain instruction sets, data, and/or other information and provide them as input into Artificial Intelligence Unit 110. In some aspects, instrumentation can be performed in source code, bytecode, compiled, interpreted, or otherwise translated code, machine code, and/or other code. In other aspects, instrumentation can be performed in various elements of a computing system such as memory, virtual machine, runtime engine/environment, operating system, compiler, interpreter, translator, processor registers, execution stack, program counter, and/or other elements. In yet other aspects, instrumentation can be performed in various abstraction layers of a computing system such as in software layer (i.e. Application Program 18, etc.), in virtual machine (if VM is used), in operating system, in Processor 11, and/or in other layers or areas that may exist in a particular computing system implementation. In yet other aspects, instrumentation can be performed at various time periods in an application's execution such as source code write time, compile time, interpretation time, translation time, linking time, loading time, runtime, and/or other time periods. In yet other aspects, instrumentation can be performed at various granularities or code segments such as some or all lines of code, some or all statements, some or all instructions or instruction sets, some or all basic blocks, some or all functions/routines/subroutines, and/or some or all other code segments.

In some embodiments, Application Program 18 can be automatically instrumented. For example, Acquisition Interface 120 can access Application Program's 18 source code, bytecode, or machine code and select instrumentation points of interest. Selecting instrumentation points may include finding locations in the source code, bytecode, or machine code corresponding to function calls, function entries, function exits, object creations, object destructions, event handler calls, new lines (i.e. to instrument all lines of code, etc.), thread creations, throws, and/or other points of interest. Instrumentation code can then be inserted at the instrumentation points of interest to output Application Program's 18 instruction sets, data, and/or other information. In response to executing instrumentation code, Application Program's 18 instruction sets, data, and/or other information may be received by Acquisition Interface 120. In some aspects, Application Program's 18 source code, bytecode, or machine code can be dynamically instrumented. For example, instrumentation code can be dynamically inserted into Application Program 18 at runtime.

In other embodiments, Application Program 18 can be manually instrumented. In one example, a programmer can instrument a function call by placing an instrumenting instruction immediately after the function call as in the following example.

```
Device1.moveForward(12);
traceApplication('Device1.moveForward(12);');
```

In another example, an instrumenting instruction can be placed immediately before the function call, or at the beginning, end, or anywhere within the function itself. A programmer may instrument all function calls or only function calls of interest. In a further example, a programmer can instrument all lines of code or only code lines of interest. In a further example, a programmer can instrument other elements utilized or implemented within Application Program 18 such as objects and/or any of their functions, data structures and/or any of their functions, event handlers and/or any of their functions, threads and/or any of their functions, and/or other elements or functions. Similar instrumentation as in the preceding examples can be performed automatically or dynamically. In some designs where manual code instrumentation is utilized, Acquisition Interface 120 can optionally be omitted and Application Program's 18 instruction sets, data, and/or other information may be transmitted directly to Artificial

Intelligence Unit 110.

In some embodiments, DCADO Unit 100 can be selective in learning instruction sets, data, and/or other information to those implemented, utilized, or related to an object, data structure, repository, thread, function, and/or other element of Application Program 18. In some aspects, Acquisition Interface 120 can obtain Application Program's 18 instruction sets, data, and/or other information implemented, utilized, or related to a certain object in an object oriented Application Program 18.

In some embodiments, various computing systems and/or platforms may provide native tools for obtaining instruction sets, data, and/or other information. Also, independent vendors may provide portable tools with similar functionalities that can be utilized across different computing systems and/or platforms. These native and portable tools may provide a wide range of functionalities to obtain runtime and other information such as instrumentation, tracing or profiling, logging application or system messages, outputting custom text messages, outputting objects or data structures, outputting functions/routines/subroutines or their invocations, outputting variable or parameter values, outputting thread or process behaviors, outputting call or other stacks, outputting processor registers, providing runtime memory access, providing inputs and/or outputs, performing live application monitoring, and/or other capabilities. One of ordinary skill in art will understand that, while all possible variations of the techniques to obtain instruction sets, data, and/or other information are too voluminous to describe, these techniques are within the scope of this disclosure.

In one example, obtaining instruction sets, data, and/or other information can be implemented through the .NET platform's native tools for application tracing or profiling such as System.Diagnostics.Trace, System.Diagnostics.Debug, and System.Diagnostics.TraceSource classes for tracing execution flow, and System.Diagnostics.Process, System.Diagnostics.EventLog, and System.Diagnostics.PerformanceCounter classes for profiling code, accessing local and remote processes, starting and stopping system processes, and interacting with Windows event logs, etc. For instance, a set of trace switches can be created that output an application's information. The switches can be configured using the .config file. For a Web application, this may typically be Web.config file associated with the project. In a Windows application, this file may typically be named applicationName.exe.config. Trace code can be added to application code automatically or manually as previously described. Appropriate listener can be created where the trace output is received. Trace code may output trace messages to a specific target such as a file, a log, a database, an object, a data structure, and/or other repository or system. Acquisition Interface 120 or Artificial Intelligence Unit 110 can then read or obtain the trace information from these targets. In some aspects, trace code may output trace messages directly to Acquisition Interface 120. In other aspects, trace code may output trace messages directly to Artificial Intelligence Unit 110. In the case of outputting trace messages to Acquisition Interface 120 or directly to Artificial Intelligence Unit 110, custom listeners can be built to accommodate these specific targets. Other platforms, tools, and/or techniques can provide equivalent or similar functionalities as the above described ones.

In another example, obtaining instruction sets, data, and/or other information can be implemented through the .NET platform's Profiling API that can be used to create a custom profiler application for tracing, monitoring, interfacing with, and/or managing a profiled application. The Profiling API provides an interface that includes methods to notify the profiler of events in the profiled application. The Profiling API may also provide an interface to enable the profiler to call back into the profiled application to obtain information about the state of the profiled

application. The Profiling API may further provide call stack profiling functionalities. Call stack (also referred to as execution stack, control stack, runtime stack, machine stack, the stack, etc.) includes a data structure that can store information about active subroutines of an application. The Profiling API may provide a stack snapshot method, which enables a trace of the stack at a particular point in time. The Profiling API may also provide a shadow stack method, which tracks the call stack at every instant. A shadow stack can obtain function arguments, return values, and information about generic instantiations. A function such as FunctionEnter can be utilized to notify the profiler that control is being passed to a function and can provide information about the stack frame and function arguments. A function such as FunctionLeave can be utilized to notify the profiler that a function is about to return to the caller and can provide information about the stack frame and function return value. An alternative to call stack profiling includes call stack sampling in which the profiler can periodically examine the stack. In some aspects, the Profiling API enables the profiler to change the in-memory code stream for a routine before it is just-in-time (JIT) compiled where the profiler can dynamically add instrumentation code to all or particular routines of interest. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through Java platform's APIs for application tracing or profiling such as Java Virtual Machine Profiling Interface (JVMPI), Java Virtual Machine Tool Interface (JVMTI), and/or other APIs or tools. These APIs can be used for instrumentation of an application, for notification of Java Virtual Machine (VM) events, and/or other functionalities. One of the tracing or profiling techniques that can be utilized includes bytecode instrumentation. The profiler can insert bytecodes into all or some of the classes. In application execution profiling, for example, these bytecodes may include methodEntry and methodExit calls. In memory profiling, for example, the bytecodes may be inserted after each new or after each constructor. In some aspects, insertion of instrumentation bytecode can be performed either by a post-compiler or a custom class loader. An alternative to bytecode instrumentation includes monitoring events generated by the JVMPI or JVMTI interfaces. Both APIs can generate events for method entry/exit, object allocation, and/or other events. In some aspects, JVMTI can be utilized for dynamic bytecode instrumentation where insertion of instrumentation bytecodes is performed at runtime. The profiler may insert the necessary instrumentation when a selected class is invoked in an application. This can be accomplished using the JVMTI's redefineClasses method, for example. This approach also enables changing of the level of profiling as the application is running. If needed, these changes can be made adaptively without restarting the application. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through JVMTI's programming interface that enables creation of software agents that can monitor and control a Java application. An agent may use the functionality of the interface to register for notification of events as they occur in the application, and to query and control the application. A JVMTI agent may use JVMTI functions to extract information from a Java application. A JVMTI agent can be utilized to obtain an application's runtime information such as method calls, memory allocation, CPU utilization, lock contention, and/or other information. JVMTI may include functions to obtain information about variables, fields, methods, classes, and/or other information. JVMTI may also provide notification for numerous events such as method entry and exit, exception, field access and modification, thread start and end, and/or other events. Examples of JVMTI built-in methods include GetMethodName to obtain the name of an invoked method, GetThreadInfo to obtain information for a specific

thread, GetClassSignature to obtain information about the class of an object, GetStackTrace to obtain information about the stack including information about stack frames, and/or other methods. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through java.lang.Runtime class that provides an interface for application tracing or profiling. Examples of methods provided in java.lang.Runtime that can be used to obtain an application's instruction sets, data, and/or other information include tracemethodcalls, traceinstructions, and/or other methods. These methods prompt the Java Virtual Machine to output trace information for a method or instruction in the virtual machine as it is executed. The destination of trace output may be system dependent and include a file, a listener, and/or other destinations where Acquisition Interface 120, Artificial Intelligence Unit 110, and/or other disclosed elements can access needed information. In addition to tracing or profiling tools native to their respective computing systems and/or platforms, many independent tools exist that provide tracing or profiling functionalities on more than one computing system and/or platform. Examples of these tools include Pin, DynamoRIO, KernInst, DynInst, Kprobes, OpenPAT, DTrace, SystemTap, and/or others. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through logging tools of the platform and/or operating system on which an application runs. Some logging tools may include nearly full feature sets of the tracing or profiling tools previously described. In one example, Visual Basic enables logging of runtime messages through its Microsoft.VisualBasic.Logging namespace that provides a log listener where the log listener may direct logging output to a file and/or other target. In another example, Java enables logging through its java.util.logging class. In some aspects, obtaining an application's instruction sets, data, and/or other information can be implemented through logging capabilities of the operating system on which an application runs. For example, Windows NT features centralized log service that applications and operating-system components can utilize to report their events including any messages. Windows NT provides functionalities for system, application, security, and/or other logging. An application log may include events logged by applications. Windows NT, for example, may include support for defining an event source (i.e. application that created the event, etc.). Windows Vista, for example, supports a structured XML log-format and designated log types to allow applications to more precisely log events and to help interpret the events. Examples of different types of event logs include administrative, operational, analytic, debug, and/or other log types including any of their subcategories. Examples of event attributes that can be utilized include eventID, level, task, opcode, keywords, and/or other event attributes. Windows wevtutil tool enables access to events, their structures, registered event publishers, and/or their configuration even before the events are fired. Wevtutil supports capabilities such as retrieval of the names of all logs on a computing device; retrieval of configuration information for a specific log; retrieval of event publishers on a computing device; reading events from an event log, from a log file, or using a structured query; exporting events from an event log, from a log file, or using a structured query to a specific target; and/or other capabilities. Operating system logs can be utilized solely if they contain sufficient information on an application's instruction sets, data, and/or other information. Alternatively, operating system logs can be utilized in combination with another source of information (i.e. trace information, call stack, processor registers, memory, etc.) to reconstruct the application's instruction sets, data, and/or other information needed for Artificial Intelligence Unit 110 and/or other elements. In

addition to logging capabilities native to their respective platforms and/or operating systems, many independent tools exist that provide logging on different platforms and/or operating systems. Examples of these tools include Log4j, Logback, SmartInspect, NLog, log4net, Microsoft Enterprise Library, ObjectGuy Framework, and/or others. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through tracing or profiling the operating system on which an application runs. As in tracing or profiling an application, one of the techniques that can be utilized includes adding instrumentation code to the operating system's source code. Such instrumentation code can be added to the operating system's source code before kernel compilation or recompilation, for instance. This type of instrumentation may involve defining or finding locations in the operating system's source code where instrumentation code may be inserted. Kernel instrumentation can also be performed without the need for kernel recompilation or rebooting. In some aspects, instrumentation code can be added at locations of interest through binary rewriting of compiled kernel code. In other aspects, kernel instrumentation can be performed dynamically where instrumentation code is added and/or removed where needed at runtime. For instance, dynamic instrumentation may overwrite kernel code with a branch instruction that redirects execution to instrumentation code or instrumentation routine. In yet other aspects, kernel instrumentation can be performed using just-in-time (JIT) dynamic instrumentation where execution may be redirected to a copy of kernel's code segment that includes instrumentation code. This type of instrumentation may include a JIT compiler and creation of a copy of the original code segment having instrumentation code or calls to instrumentation routines embedded into the original code segment. Instrumentation of the operating system may enable total system visibility including visibility into an application's behavior by enabling generation of low level trace information. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through tracing or profiling the processor on which an application runs. For example, some Intel processors provide Intel Processor Trace (i.e. Intel PT, etc.), a low-level tracing feature that enables recording executed instruction sets, and/or other data or information of one or more applications. Intel PT is facilitated by the Processor Trace Decoder Library along with its related tools. Intel PT is a low-overhead execution tracing feature that records information about application execution on each hardware thread using dedicated hardware facilities. The recorded execution/trace information is collected in data packets that can be buffered internally before being sent to a memory subsystem or another system or element (i.e. Acquisition Interface 120, Artificial Intelligence Unit 110, etc.). Intel PT also enables navigating the recorded execution/trace information via reverse stepping commands. Intel PT can be included in an operating system's core files and provided as a feature of the operating system. Intel PT can trace globally some or all applications running on an operating system. Acquisition Interface 120 or Artificial Intelligence Unit 110 can read or obtain the recorded execution/trace information from Intel PT. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through branch tracing or profiling. Branch tracing may include an abbreviated instruction trace in which only the successful branch instruction sets are traced or recorded. Branch tracing can be implemented through utilizing dedicated processor commands, for example. Executed branches may be saved into special branch trace store area of

memory. With the availability and reference to a compiler listing of the application together with branch trace information, a full path of executed instruction sets can be reconstructed. The full path can also be reconstructed with a memory dump (containing the program storage) and branch trace information. In some aspects, branch tracing can be utilized for pre-learning or automated learning of an application's instruction sets, data, and/or other information where a number of application simulations (i.e. simulations of likely/common operations, etc.) are performed. As such, the application's operation can be learned automatically saving the time that would be needed to learn the application's operation directed by a user. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through assembly language. Assembly language is a low-level programming language for a computer or other programmable device in which there is a strong correlation between the language and the architecture's machine instruction sets. Syntax, addressing modes, operands, and/or other elements of an assembly language instruction set may translate directly into numeric (i.e. binary, etc.) representations of that particular instruction set. Because of this direct relationship with the architecture's machine instruction sets, assembly language can be a powerful tool for tracing or profiling an application's execution in processor registers, memory, and/or other computing system components. For example, using assembly language, memory locations of a loaded application can be accessed, instrumented, and/or otherwise manipulated. In some aspects, assembly language can be used to rewrite or overwrite original in-memory instruction sets of an application with instrumentation instruction sets. In other aspects, assembly language can be used to redirect application's execution to instrumentation routine/subroutine or other code segment elsewhere in memory by inserting a jump into the application's in-memory code, by redirecting program counter, or by other techniques. Some operating systems may implement protection from changes to applications loaded into memory. Operating system, processor, or other low level commands such as Linux mprotect command or similar commands in other operating systems may be used to unprotect the protected locations in memory before the change. In yet other aspects, assembly language can be used to obtain instruction sets, data, and/or other information through accessing and/or reading instruction register, program counter, other processor registers, memory locations, and/or other components of a computing system. In yet other aspects, high-level programming languages may call or execute an external assembly language program to facilitate obtaining instruction sets, data, and/or other information as previously described. In yet other aspects, relatively low-level programming languages such as C may allow embedding assembly language directly in their source code such as, for example, using asm keyword of C. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, it may be sufficient to obtain user or other inputs, variables, parameters, and/or other data in some procedural, simple object oriented, or other applications. In one instance, a simple procedural application executes a sequence of instruction sets until the end of the program. During its execution, the application may receive user or other input, store the input in a variable, and perform calculations using the variable to reach a result. The value of the variable can be obtained or traced. In another instance, a more complex procedural application comprises one or more functions/routines/subroutines each of which may include a sequence of instruction sets. The application may execute a main sequence of instruction sets with a branch to a function/routine/subroutine. During its execution, the application may receive user or other input, store the input in a

variable, and pass the variable as a parameter to the function/routine/subroutine. The function/routine/subroutine may perform calculations using the parameter and return a value that the rest of the application can use to reach a result. The value of the variable or parameter passed to the function/routine/subroutine, and/or return value can be obtained or traced. Values of user or other inputs, variables, parameters, and/or other items of interest can be obtained through previously described tracing, instrumentation, and/or other techniques. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

Referring to Fig. 5, in yet another example, obtaining instruction sets, data, and/or other information may be implemented through tracing, profiling, or sampling of instruction sets or data in processor registers, memory, or other computing system components where instruction sets, data, and/or other information may be stored or utilized. For example, Instruction Register 212 may be part of Processor 11 and it may store the instruction set currently being executed or decoded. In some processors, Program Counter 211 (also referred to as instruction pointer, instruction address register, instruction counter, or part of instruction sequencer) may be incremented after fetching an instruction set, and it may hold or point to the memory address of the next instruction set to be executed. In a processor where the incrementation precedes the fetch, Program Counter 211 may point to the current instruction set being executed. In the instruction cycle, an instruction set may be loaded into Instruction Register 212 after Processor 11 fetches it from location in Memory 12 pointed to by Program Counter 211. Instruction Register 212 may hold the instruction set while it is decoded by Instruction Decoder 213, prepared, and executed. In some aspects, data (i.e. operands, etc.) needed for instruction set execution may be loaded from Memory 12 into a register within Register Array 214. In other aspects, the data may be loaded directly into Arithmetic Logic Unit 215. For instance, as instruction sets pass through Instruction Register 212 during application execution, they may be transmitted to Acquisition Interface 120 as shown. Examples of the steps in execution of a machine instruction set may include decoding the opcode (i.e. portion of a machine instruction set that may specify the operation to be performed), determining where the operands may be located (depending on architecture, operands may be in registers, the stack, memory, I/O ports, etc.), retrieving the operands, allocating processor resources to execute the instruction set (needed in some types of processors), performing the operation indicated by the instruction set, saving the results of execution, and/or other execution steps. Examples of the types of machine instruction sets that can be utilized include arithmetic, data handling, logical, program control, as well as special and/or other instruction set types. In addition to the ones described or shown, examples of other computing system or processor components that can be used during an instruction cycle include memory address register (MAR) that may hold the address of a memory block to be read from or written to; memory data register (MDR) that may hold data fetched from memory or data waiting to be stored in memory; data registers that may hold numeric values, characters, small bit arrays, or other data; address registers that may hold addresses used by instruction sets that indirectly access memory; general purpose registers (GPRs) that may store both data and addresses; conditional registers that may hold truth values often used to determine whether some instruction set should or should not be executed; floating point registers (FPRs) that may store floating point numbers; constant registers that may hold read-only values such as zero, one, or pi; special purpose registers (SPRs) such as status register, program counter, or stack pointer that may hold information on program state; machine-specific registers that may store data and settings related to a particular processor; Register Array 214 that may include an array of any number of processor registers; Arithmetic Logic Unit 215 that may perform arithmetic and logic operations; control unit that may direct processor's operation;

and/or other circuits or components. Tracing, profiling, or sampling of processor registers, memory, or other computing system components can be implemented in a program, combination of hardware and program, or purely hardware system. Dedicated hardware may be built to perform tracing, profiling, or sampling of processor registers or any computing system components with marginal or no impact to computing overhead.

One of ordinary skill in art will recognize that Fig. 5 depicts one of many implementations of processor or computing system components, and that various additional components can be included, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate implementations. Processor or computing system components may be arranged or connected differently in alternate implementations. Processor or computing system components may also be connected with external elements using various connections. For instance, the connection between Instruction Register 212 and Acquisition Interface 120 may include any number or types of connections such as, for example, a dedicated connection for each bit of Instruction Register 212 (i.e. 32 connections for a 32 bit Instruction Register 212, etc.). Any of the described or other connections or interfaces may be implemented among any processor or computing system components and Acquisition Interface 120 or other elements.

Referring to Figs. 6A-6B, in yet another example, obtaining instruction sets, data, and/or other information may be implemented through tracing, profiling, or sampling of Logic Circuit 250. While Processor 11 includes any type or embodiment of logic circuit, Logic Circuit 250 is described separately here to offer additional detail on its functioning. Some Devices 98 may not need the processing capabilities of an entire Processor 11, but instead a more tailored Logic Circuit 250. Examples of such Devices 98 include home appliances, audio or video electronics, vehicle systems, toys, industrial machines, robots, and/or others. Logic Circuit 250 comprises the functionality for performing logic operations. Logic Circuit 250 comprises the functionality for performing logic operations using the circuit's inputs and producing outputs based on the logic operations performed on the inputs. Logic Circuit 250 may generally be implemented using transistors, diodes, and/or other electronic switches, but can also be constructed using vacuum tubes, electromagnetic relays (relay logic), fluidic logic, pneumatic logic, optics, molecules, or even mechanical elements. In some aspects, Logic Circuit 250 may be or include a microcontroller, field-programmable gate array (FPGA), application-specific integrated circuit (ASIC), and/or other computing circuit or device. In other aspects, Logic Circuit 250 may be or include any circuit or device comprising one or more logic gates, one or more transistors, one or more switches, and/or one or more other logic components. In further aspects, Logic Circuit 250 may be or include any integrated or other circuit or device that can perform logic operations. Logic may generally refer to Boolean logic utilized in binary operations, but other logics can also be used. Input into Logic Circuit 250 may include or refer to a value inputted into the Logic Circuit 250, therefore, these terms may be used interchangeably herein depending on context. In one example, Logic Circuit 250 may perform some logic operations using four input values and produce two output values. As the four input values are delivered to or received by Logic Circuit 250, they may be obtained by Acquisition Interface 120 through the four hardwired connections as shown in Fig. 6A. In another example, Logic Circuit 250 may perform some logic operations using four input values and produce two output values. As the two output values are generated by or transmitted out of Logic Circuit 250, they may be obtained by Acquisition Interface 120 through the two hardwired connections as shown in Fig. 6B. In a further example, instead of or in addition to obtaining input and/or output values of Logic Circuit 250, the state of Logic Circuit 250 may be obtained by reading or accessing values from one or more Logic Circuit's 250 internal

components such as registers, memories, buses, and/or others (i.e. similar to the previously described tracing, profiling, and/or sampling of Processor 11 components, etc.). Tracing, profiling, or sampling of Logic Circuit 250 can be implemented in a program, combination of hardware and program, or purely hardware system. Dedicated hardware may be built to perform tracing, profiling, or sampling of Logic Circuit 250 with marginal or no impact to computing overhead. Any of the elements and/or techniques for tracing, profiling, or sampling of Logic Circuit 250 can similarly be implemented with Processor 11 and/or other processing elements. In some designs, DCADO Unit 100 may include clamps and/or other elements to attach DCADO Unit 100 to inputs (i.e. input wires, etc.) into and/or outputs (i.e. output wires, etc.) from Logic Circuit 250. Such clamps and/or attachment elements enable seamless attachment of DCADO Unit 100 to any circuit or computing device without the need to redesign or alter the circuit or computing device.

In some embodiments, DCADO Unit 100 may learn input values directly from an actuator (previously described, not shown). For example, Logic Circuit 250 or other processing element may control an actuator that enables Device 98 to perform mechanical, physical, and/or other operations. An actuator may receive one or more input values or control signals from Logic Circuit 250 or other processing element directing the actuator to perform specific operations. As one or more input values or control signals are delivered to or received by the actuator, they may be obtained by Acquisition Interface 120 as previously described with respect to obtaining input values of Logic Circuit 250. Specifically, for instance, one or more input values or control signals of an actuator may be obtained by Acquisition Interface 120 via hardwired or other connections.

One of ordinary skill in art will recognize that Figs. 6A-6B depict one of many implementations of Logic Circuit 250 and that any number of input and/or output values can be utilized in alternate implementations. One of ordinary skill in art will also recognize that Logic Circuit 250 may include any number and/or combination of logic components to implement any logic operations.

Other additional techniques or elements may be utilized as needed for obtaining instruction sets, data, and/or other information, or some of the disclosed techniques or elements may be excluded, or a combination thereof may be utilized in alternate embodiments.

Referring to Figs. 7A-7E, some embodiments of Instruction Sets 526 are illustrated. In some aspects, Instruction Set 526 includes one or more instructions or commands of Application Program 18. For example, Instruction Set 526 may include one or more instructions or commands of a high-level programming language such as Java or SQL, a low-level language such as assembly or machine language, an intermediate language or construct such as bytecode, and/or any other language or construct. In other aspects, Instruction Set 526 includes one or more inputs into and/or outputs from Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element. In further aspects, Instruction Set 526 includes one or more values or states of registers and/or other components of Logic Circuit 250, Processor 11, and/or other processing element. In general, Instruction Set 526 may include one or more instructions, commands, keywords, symbols (i.e. parentheses, brackets, commas, semicolons, etc.), operators (i.e. =, <, >, etc.), variables, values, objects, data structures, functions (i.e. Function1(), FIRST(), MIN(), SQRT(), etc.), parameters, states, signals, inputs, outputs, characters, digits, references thereto, and/or other components for performing an operation.

In an embodiment shown in Fig. 7A, Instruction Set 526 includes code of a high-level programming language (i.e. Java, C++, etc.) comprising the following function call construct: Function1 (Parameter1, Parameter2,

Parameter3, ...). An example of a function call applying the above construct includes the following Instruction Set 526: moveTo(Device1, 14, 8). The function or reference thereto "moveTo(Device1, 14, 8)" may be an Instruction Set 526 directing Device1 to move to a location with coordinates 14 and 8, for example. In another embodiment shown in Fig. 7B, Instruction Set 526 includes structured query language (SQL). In a further embodiment shown in Fig. 7C, Instruction Set 526 includes bytecode (i.e. Java bytecode, Python bytecode, CLR bytecode, etc.). In a further embodiment shown in Fig. 7D, Instruction Set 526 includes assembly code. In a further embodiment shown in Fig. 7E, Instruction Set 526 includes machine code.

Referring to Figs. 8A-8B, some embodiments of Extra Information 527 (also referred to as Extra Info 527) are illustrated. In an embodiment shown in Fig. 8A, Collection of Object Representations 525 may include or be associated with Extra Info 527. In an embodiment shown in Fig. 8B, Instruction Set 526 may include or be associated with Extra Info 527.

Extra Info 527 comprises the functionality for storing any information useful in comparisons or decision making performed in autonomous device operation, and/or other functionalities. One or more Extra Infos 527 can be stored in, appended to, or associated with a Collection of Object Representations 525, Instruction Set 526, and/or other element. In some embodiments, the system can obtain Extra Info 527 at a time of creating or generating Collection of Object Representations 525. In other embodiments, the system can obtain Extra Info 527 at a time of acquiring Instruction Set 526. In general, Extra Info 527 can be obtained at any time. Examples of Extra Info 527 include time information, location information, computed information, contextual information, and/or other information. Any information can be utilized that can provide information for enhanced comparisons or decision making performed in autonomous device operation. Which information is utilized and/or stored in Extra Info 527 can be set by a user, by DCADO system administrator, or automatically by the system. Extra Info 527 may include or be referred to as contextual information, and vice versa. Therefore, these terms may be used interchangeably herein depending on context.

In some aspects, time information (i.e. time stamp, etc.) can be utilized and/or stored in Extra Info 527. Time information can be useful in comparisons or decision making performed in autonomous device operation related to a specific time period as Device 98 may be required to perform specific operations at certain parts of day, month, year, and/or other time periods. Time information can be obtained from the system clock, online clock, oscillator, or other time source. In general, Extra Info 527 may include time information related to when Device 98 performed an operation. In other aspects, location information (i.e. coordinates, distance/angle from a known point, address, etc.) can be utilized and/or stored in Extra Info 527. Location information can be useful in comparisons or decision making performed in autonomous device operation related to a specific place as Device 98 may be required to perform specific operations at certain places. Location information can be obtained from a positioning system (i.e. radio signal triangulation, GPS capabilities, etc.), sensors, and/or other location system. In general, Extra Info 527 may include location information related to where Device 98 performed an operation. In further aspects, computed information can be utilized and/or stored in Extra Info 527. Computed information can be useful in comparisons or decision making performed in autonomous device operation where information can be calculated, inferred, or derived from other available information. DCADO Unit 100 and/or other disclosed elements may include computational functionalities to create Extra Info 527 by performing calculations or inferences using other information. In one example, Device's 98 speed can be computed or estimated from Device's 98 location and/or time

information. In another example, Device's 98 bearing (i.e. angle or direction of movement, etc.) can be computed or estimated from Device's 98 location information by utilizing Pythagorean theorem, trigonometry, and/or other theorems, formulas, or disciplines. In a further example, speeds, bearings, distances, and/or other properties of objects around Device 98 can similarly be computed or inferred using known information. In further aspects, observed information can be utilized and/or stored in Extra Info 527. In further aspects, other information can be utilized and/or stored in Extra Info 527. Examples of such other information include user specific information (i.e. skill level, age, gender, etc.), group user information (i.e. access level, etc.), version of Application Program 18, type of Application Program 18, type of Processor 11, type of Logic Circuit 250, type of Device 98, and/or other information.

Referring to Fig. 9, an embodiment where DCADO Unit 100 is part of or operating on Processor 11 is illustrated. In one example, DCADO Unit 100 may be a hardware element or circuit embedded or built into Processor 11. In another example, DCADO Unit 100 may be a program operating on Processor 11.

Referring to Fig. 10, an embodiment where DCADO Unit 100 resides on Server 96 accessible over Network 95 is illustrated. Any number of Devices 98 may connect to such remote DCADO Unit 100 and the remote DCADO Unit 100 may learn their operations in circumstances including objects with various properties. In turn, any number of Devices 98 can utilize the remote DCADO Unit 100 for autonomous operation in circumstances including objects with various properties. A remote DCADO Unit 100 can be offered as a network service (i.e. online application, etc.). In some aspects, a remote DCADO Unit 100 (i.e. global DCADO Unit 100, etc.) may reside on the Internet and be available to all the world's Devices 98 configured to transmit their operations in circumstances including objects with various properties and/or configured to utilize the remote DCADO Unit 100 for autonomous operation in circumstances including objects with various properties. For example, multiple operators (i.e. Users 50, etc.) may operate their Devices 98 where the Devices 98 may be configured to transmit their operations in circumstances including objects with various properties to a remote DCADO Unit 100. Such remote DCADO Unit 100 enables learning of the operators' collective knowledge of operating Device 98 in circumstances including objects with various properties. Server 96 may be or include any type or form of a remote computing device such as an application server, a network service server, a cloud server, a cloud, and/or other remote computing device. Server 96 may include any features, functionalities, and embodiments of the previously described Computing Device 70. It should be understood that Server 96 does not have to be a separate computing device and that Server 96, its elements, or its functionalities can be implemented on Device 98. Network 95 may include various networks, connection types, protocols, interfaces, APIs, and/or other elements or techniques known in art all of which are within the scope of this disclosure. Any of the previously described networks, network or connection types, networking interfaces, and/or other networking elements or techniques can similarly be utilized. Any of the disclosed elements may reside on Server 96 in alternate implementations. In one example, Artificial Intelligence Unit 110 can reside on Server 96 and Acquisition Interface 120 and/or Modification Interface 130 can reside on Device 98. In another example, Knowledgebase 530 can reside on Server 96 and the rest of the elements of DCADO Unit 100 can reside on Device 98. Any other combination of local and remote elements can be implemented.

Referring to Fig. 11, an embodiment of learning and/or using Remote Device's 97 circumstances for autonomous Remote Device 97 operation is illustrated. In such embodiments, in addition to providing input into Object Processing Unit 93 for learning functionalities herein, Sensor 92 (i.e. Camera 92a, Radar 92d, Sonar 92e, etc.) can provide input into Display 21 or other device for User's 50 perception of Remote Device's 97 surrounding.

As User 50 operates Remote Device 97, DCADO Unit 100 may learn Remote Device's 97 operation in circumstances including objects with various properties. Such embodiments can be utilized in any situation where one device controls (i.e. remote controls, etc.) another device, any situation where some or all of the processing is on one device and sensor capabilities are on another device, and/or other situations. In one example, a drone controlling device (i.e. Device 98, etc.) may send control signals to operate a drone (i.e. Remote Device 97, etc.) and receive information on the drone's surrounding from Sensor 92 on the drone. In another example, a robot controlling device (i.e. Device 98, etc.) may send control signals to operate a robot (i.e. Remote Device 97, etc.) and receive information on the robot's surrounding from Sensor 92 on the robot. Any of the disclosed elements in addition to Sensor 92 may reside on Remote Device 97 in alternate implementations.

Referring to Fig. 12, an embodiment of Artificial Intelligence Unit 110 is illustrated. Artificial Intelligence Unit 110 comprises interconnected Knowledge Structuring Unit 520, Knowledgebase 530, Decision-making Unit 540, and Confirmation Unit 550. Other additional elements can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate embodiments.

Artificial Intelligence Unit 110 comprises the functionality for learning Device's 98 operation in circumstances including objects with various properties. Artificial Intelligence Unit 110 comprises the functionality for learning one or more collections of object representations correlated with any instruction sets, data, and/or other information. In some aspects, Artificial Intelligence Unit 110 comprises the functionality for learning one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527. In other aspects, Artificial Intelligence Unit 110 comprises the functionality for learning one or more Collections of Object Representations 525 some of which may not be correlated with any Instruction Sets 526 and/or Extra Info 527. Further, Artificial Intelligence Unit 110 comprises the functionality for anticipating Device's 98 operation in circumstances including objects with various properties. Artificial Intelligence Unit 110 comprises the functionality for anticipating one or more instruction sets, data, and/or other information. Artificial Intelligence Unit 110 comprises the functionality for anticipating one or more Instruction Sets 526 based on one or more incoming Collections of Object Representations 525. Artificial Intelligence Unit 110 comprises the functionality for anticipating one or more Instruction Sets 526 to be used or executed in Device's 98 autonomous operation. Artificial Intelligence Unit 110 also comprises other disclosed functionalities.

Knowledge Structuring Unit 520, Knowledgebase 530, and Decision-making Unit 540 are described later.

Confirmation Unit 550 comprises the functionality for confirming, modifying, evaluating (i.e. rating, etc.), and/or canceling one or more anticipatory Instruction Sets 526, and/or other functionalities. Confirmation Unit 550 is an optional element that can be omitted depending on implementation. In some embodiments, Confirmation Unit 550 can serve as a means of confirming anticipatory Instruction Sets 526. For example, Decision-making Unit 540 may determine one or more anticipatory Instruction Sets 526 and provide them to User 50 for confirmation. User 50 may be provided with an interface (i.e. graphical user interface, selectable list of anticipatory Instruction Sets 526, etc.) to approve or confirm execution of the anticipatory Instruction Sets 526. In some aspects, Confirmation Unit 550 can automate User 50 confirmation. In one example, if one or more incoming Collections of Object Representations 525 from Object Processing Unit 93 and one or more Collections of Object Representations 525 from a Knowledge Cell 800 were found to be a perfect or highly similar match, anticipatory Instruction Sets 526 correlated with the one or more Collections of Object Representations 525 from the Knowledge Cell 800 can be

automatically executed without User's 50 confirmation. Conversely, if one or more incoming Collections of Object Representations 525 from Object Processing Unit 93 and one or more Collections of Object Representations 525 from a Knowledge Cell 800 were found to be less than a highly similar match, anticipatory Instruction Sets 526 correlated with the one or more Collections of Object Representations 525 from the Knowledge Cell 800 can be presented to User 50 for confirmation and/or modifying. Any features, functionalities, and/or embodiments of Similarity Comparison 125 (later described) can be utilized for such similarity determination. In other embodiments, Confirmation Unit 550 can serve as a means of modifying or editing anticipatory Instruction Sets 526. For example, Decision-making Unit 540 may determine one or more anticipatory Instruction Sets 526 and provide them to User 50 for modification. User 50 may be provided with an interface (i.e. graphical user interface, etc.) to modify the anticipatory Instruction Sets 526 before their execution. In further embodiments, Confirmation Unit 550 can serve as a means of evaluating or rating anticipatory Instruction Sets 526 if they matched User's 50 intended operation. For example, Decision-making Unit 540 may determine one or more anticipatory Instruction Sets 526, which the system may automatically execute. User 50 may be provided with an interface (i.e. graphical user interface, etc.) to rate (i.e. on a scale from 0 to 1, etc.) how well Decision-making Unit 540 predicted the executed anticipatory Instruction Sets 526. In some aspects, rating can be automatic and based on a particular function or method that rates how well the anticipatory Instruction Sets 526 matched the desired operation. In one example, a rating function or method can assign a higher rating to anticipatory Instruction Sets 526 that were least modified in the confirmation process. In another example, a rating function or method can assign a higher rating to anticipatory Instruction Sets 526 that were canceled least number of times by User 50. Any other automatic rating function or method can be utilized. In yet other embodiments, Confirmation Unit 550 can serve as a means of canceling anticipatory Instruction Sets 526 if they did not match User's 50 intended operation. For example, Decision-making Unit 540 may determine one or more anticipatory Instruction Sets 526, which the system may automatically execute. The system may save the state of Device 98, Processor 11 (save its register values, etc.), Logic Circuit 250, Application Program 18 (i.e. save its variables, data structures, objects, location of its current instruction, etc.), and/or other processing elements before executing anticipatory Instruction Sets 526. User 50 may be provided with an interface (i.e. graphical user interface, selectable list of prior executed anticipatory Instruction Sets 526, etc.) to cancel one or more of the prior executed anticipatory Instruction Sets 526, and restore Device 98, Processor 11, Logic Circuit 250, Application Program 18, and/or other processing elements to a prior state. In some aspects, Confirmation Unit 550 can optionally be disabled or omitted in order to provide an uninterrupted operation of Device 98, Processor 11, Logic Circuit 250, and/or Application Program 18. For example, a thermostat may be suitable for implementing the user confirmation step, whereas, a vehicle may be less suitable for implementing such interrupting step due to the real time nature of vehicle operation.

Referring to Fig. 13, an embodiment of Knowledge Structuring Unit 520 correlating individual Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527 is illustrated. Knowledge Structuring Unit 520 comprises the functionality for structuring the knowledge of a device's operation in circumstances including objects with various properties, and/or other functionalities. Knowledge Structuring Unit 520 comprises the functionality for correlating one or more Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527. Knowledge Structuring Unit 520 comprises the functionality for creating or generating Knowledge Cell 800 and storing one or more Collections of Object Representations 525 correlated with any

Instruction Sets 526 and/or Extra Info 527 into the Knowledge Cell 800. As such, Knowledge Cell 800 comprises the functionality for storing one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527. Knowledge Cell 800 includes knowledge (i.e. unit of knowledge, etc.) of how Device 98 operated in a circumstance including objects with various properties. Once created or generated, Knowledge Cells 800 can be used in/as neurons, nodes, vertices, or other elements in any of the data structures or arrangements (i.e. neural networks, graphs, sequences, etc.) used for storing the knowledge of Device's 98 operation in circumstances including objects with various properties, thereby facilitating learning functionalities herein. It should be noted that Extra Info 527 may be optionally used in some implementations to enable enhanced comparisons or decision making in autonomous device operation where applicable, and that Extra Info 527 can be omitted in alternate implementations.

In some embodiments, Knowledge Structuring Unit 520 receives one or more Collections of Object Representations 525 from Object Processing Unit 93. Knowledge Structuring Unit 520 may also receive one or more Instruction Sets 526 from Acquisition Interface 120. Knowledge Structuring Unit 520 may further receive any Extra Info 527. Although, Extra Info 527 is not shown in this and/or other figures for clarity of illustration, it should be noted that any Collection of Object Representations 525, Instruction Set 526, and/or other element may include or be associated with Extra Info 527. Knowledge Structuring Unit 520 may correlate one or more Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527. Knowledge Structuring Unit 520 may then create Knowledge Cell 800 and store the one or more Collections of Object Representations 525 correlated with Instruction Sets 526 and/or Extra Info 527 into the Knowledge Cell 800. Knowledge Cell 800 may include any data structure or arrangement that can facilitate such storing. For example, Knowledge Structuring Unit 520 may create Knowledge Cell 800ax and structure within it Collection of Object Representations 525a1 correlated with Instruction Sets 526a1-526a3 and/or any Extra Info 527 (not shown). Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax a Collection of Object Representations 525a2 correlated with Instruction Set 526a4 and/or any Extra Info 527 (not shown). Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax a Collection of Object Representations 525a3 without a correlated Instruction Set 526 and/or Extra Info 527. Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax a Collection of Object Representations 525a4 correlated with Instruction Sets 526a5-526a6 and/or any Extra Info 527 (not shown). Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax a Collection of Object Representations 525a5 without a correlated Instruction Set 526 and/or Extra Info 527. Knowledge Structuring Unit 520 may structure within Knowledge Cell 800ax additional Collections of Object Representations 525 correlated with any number (including zero [i.e. uncorrelated]) of Instruction Sets 526 and/or Extra Info 527 by following similar logic as described above.

In some embodiments, Knowledge Structuring Unit 520 may correlate a Collection of Object Representations 525 with one or more temporally corresponding Instruction Sets 526 and/or Extra Info 527. This way, Knowledge Structuring Unit 520 can structure the knowledge of Device's 98 operation at or around the time of generating Collections of Object Representations 525. Such functionality enables spontaneous or seamless learning of Device's 98 operation in circumstances including objects with various properties as Device 98 is operated in real life situations. In some designs, Knowledge Structuring Unit 520 may receive a stream of Instruction Sets 526 used or executed to effect Device's 98 operations as well as a stream of Collections of Object Representations 525 as the

operations are performed. Knowledge Structuring Unit 520 can then correlate Collections of Object Representations 525 from the stream of Collections of Object Representations 525 with temporally corresponding Instruction Sets 526 from the stream of Instruction Sets 526 and/or any Extra Info 527. Collections of Object Representations 525 without a temporally corresponding Instruction Set 526 may be uncorrelated, for instance. In some aspects, Instruction Sets 526 and/or Extra Info 527 that temporally correspond to a Collection of Object Representations 525 may include Instruction Sets 526 used and/or Extra Info 527 obtained at the time of generating the Collection of Object Representations 525. In other aspects, Instruction Sets 526 and/or Extra Info 527 that temporally correspond to a Collection of Object Representations 525 may include Instruction Sets 526 used and/or Extra Info 527 obtained within a certain time period before and/or after generating the Collection of Object Representations 525. For example, Instruction Sets 526 and/or Extra Info 527 that temporally correspond to a Collection of Object Representations 525 may include Instruction Sets 526 used and/or Extra Info 527 obtained within 50 milliseconds, 1 second, 3 seconds, 20 seconds, 1 minute, 41 minutes, 2 hours, or any other time period before and/or after generating the Collection of Object Representations 525. Such time periods can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. In other aspects, Instruction Sets 526 and/or Extra Info 527 that temporally correspond to a Collection of Object Representations 525 may include Instruction Sets 526 used and/or Extra Info 527 obtained from the time of generating the Collection of Object Representations 525 to the time of generating a next Collection of Object Representations 525. In further aspects, Instruction Sets 526 and/or Extra Info 527 that temporally correspond to a Collection of Object Representations 525 may include Instruction Sets 526 used and/or Extra Info 527 obtained from the time of generating a previous Collection of Object Representations 525 to the time of generating the Collection of Object Representations 525. Any other temporal relationship or correspondence between Collections of Object Representations 525 and correlated Instruction Sets 526 and/or Extra Info 527 can be implemented.

In some embodiments, Knowledge Structuring Unit 520 can structure the knowledge of Device's 98 operation in a circumstance including objects with various properties into any number of Knowledge Cells 800. In some aspects, Knowledge Structuring Unit 520 can structure into a Knowledge Cell 800 a single Collection of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527. In other aspects, Knowledge Structuring Unit 520 can structure into a Knowledge Cell 800 any number (i.e. 2, 4, 7, 17, 29, 87, 1415, 23891, 323674, 8132401, etc.) of Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527. In a special case, Knowledge Structuring Unit 520 can structure all Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 into a single long Knowledge Cell 800. In further aspects, Knowledge Structuring Unit 520 can structure Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 into a plurality of Knowledge Cells 800. In a special case, Knowledge Structuring Unit 520 can store periodic streams of Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 into a plurality of Knowledge Cells 800 such as hourly, daily, weekly, monthly, yearly, or other periodic Knowledge Cells 800.

In some embodiments, Device 98 may include a plurality of Sensors 92 and/or their corresponding Object Processing Units 93. In one example, multiple Sensors 92 may detect objects and/or their properties from different angles or on different sides of Device 98. In another example, one or more Sensors 92 may be placed on different

sub-devices, sub-systems, or elements of Device 98. Using multiple Sensors 92 and/or their corresponding Object Processing Units 93 may provide additional detail in learning and/or using Device's 98 circumstances for autonomous Device 98 operation. In some designs where multiple Sensors 92 and/or their corresponding Object Processing Units 93 are utilized, multiple DCADO Units 100 can also be utilized (i.e. one DCADO Unit 100 for each Sensor 92 and its corresponding Object Processing Unit 93, etc.). In such designs, Collections of Object Representations 525 can be correlated with any Instruction Sets 526 and/or Extra Info 527 as previously described. In other designs where multiple Sensors 92 and/or their corresponding Object Processing Units 93 are utilized, collective Collections of Object Representations 525 from multiple Sensors 92 and their corresponding Object Processing Units 93 can be correlated with any Instruction Sets 526 and/or Extra Info 527.

In some embodiments, Device 98 may include a plurality of Logic Circuits 250, Processors 11, Application Programs 18, and/or other processing elements. For example, each processing element may control a sub-device, sub-system, or an element of Device 98. Using multiple processing elements may provide enhanced control over Device's 98 operation. In some designs where multiple processing elements are utilized, multiple DCADO Units 100 can also be utilized (i.e. one DCADO Unit 100 for each processing element, etc.). In such designs, Collections of Object Representations 525 can be correlated with any Instruction Sets 526 and/or Extra Info 527 as previously described. In other designs where multiple processing elements are utilized, Collections of Object Representations 525 can be correlated with any collective Instruction Sets 526 and/or Extra Info 527 used or executed by a plurality of processing elements.

Any combination of the aforementioned multiple Sensors 92 and/or their corresponding Object Processing Units 93, multiple processing elements, and/or other elements can be implemented in alternate embodiments.

Referring to Fig. 14, another embodiment of Knowledge Structuring Unit 520 correlating individual Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527 is illustrated. In such embodiments, Knowledge Structuring Unit 520 may generate Knowledge Cells 800 each comprising a single Collection of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527.

Referring to Fig. 15, an embodiment of Knowledge Structuring Unit 520 correlating streams of Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527 is illustrated. For example, Knowledge Structuring Unit 520 may create Knowledge Cell 800ax and structure within it a stream of Collections of Object Representations 525a1-525an correlated with Instruction Set 526a1 and/or any Extra Info 527 (not shown). Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax a stream of Collections of Object Representations 525b1-525bn correlated with Instruction Sets 526a2-526a4 and/or and Extra Info 527 (not shown). Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax a stream of Collections of Object Representations 525c1-525cn without correlated Instruction Sets 526 and/or Extra Info 527. Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax a stream of Collections of Object Representations 525d1-525dn correlated with Instruction Sets 526a5-526a6 and/or any Extra Info 527 (not shown). Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax additional streams of Collections of Object Representations 525 correlated with any number (including zero [i.e. uncorrelated]) of Instruction Sets 526 and/or Extra Info 527 by following similar logic as described above. The number of Collections of Object Representations 525 in some or all streams of Collections of Object Representations 525a1-525an, 525b1-525bn, etc. may be equal or different. It should be noted that n or other such letters or indicia may follow the sequence and/or context where

they are indicated. Also, a same letter or indicia such as n may represent a different number in different elements of a drawing.

Referring to Fig. 16, another embodiment of Knowledge Structuring Unit 520 correlating streams of Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527 is illustrated. In such embodiments, Knowledge Structuring Unit 520 may generate Knowledge Cells 800 each comprising a single stream of Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527.

Knowledgebase 530 comprises the functionality for storing the knowledge of a device's operation in circumstances including objects with various properties, and/or other functionalities. Knowledgebase 530 comprises the functionality for storing one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527. Knowledgebase 530 comprises the functionality for storing one or more Knowledge Cells 800 each including one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527. In some aspects, Collections of Object Representations 525 correlated with Instruction Sets 526 and/or Extra Info 527 can be stored directly within Knowledgebase 530 without using Knowledge Cells 800 as the intermediary data structures. In some embodiments, Knowledgebase 530 may be or include Neural Network 530a (later described). In other embodiments, Knowledgebase 530 may be or include Graph 530b (later described). In further embodiments, Knowledgebase 530 may be or include Collection of Sequences 530c (later described). In further embodiments, Knowledgebase 530 may be or include Sequence 533 (later described). In further embodiments, Knowledgebase 530 may be or include Collection of Knowledge Cells 530d (later described). In general, Knowledgebase 530 may be or include any data structure or arrangement capable of storing the knowledge of a device's operation in circumstances including objects with various properties. Knowledgebase 530 may reside locally on Device 98, or remotely (i.e. remote Knowledgebase 530, etc.) on a remote computing device (i.e. server, cloud, etc.) accessible over a network or an interface.

In some embodiments, Knowledgebase 530 from one Device 98 or DCADO Unit 100 can be transferred to one or more other Devices 98 or DCADO Units 100. Therefore, the knowledge of Device's 98 operation in circumstances including objects with various properties learned on one Device 98 or DCADO Unit 100 can be transferred to one or more other Devices 98 or DCADO Units 100. In one example, Knowledgebase 530 can be copied or downloaded to a file or other repository from one Device 98 or DCADO Unit 100 and loaded or inserted into another Device 98 or DCADO Unit 100. In another example, Knowledgebase 530 from one Device 98 or DCADO Unit 100 can be available on a server accessible by other Devices 98 or DCADO Units 100 over a network or an interface. Once loaded into or accessed by a receiving Device 98 or DCADO Unit 100, the receiving Device 98 or DCADO Unit 100 can then implement the knowledge of Device's 98 operation in circumstances including objects with various properties learned on the originating Device 98 or DCADO Unit 100.

In some embodiments, multiple Knowledgebases 530 (i.e. Knowledgebases 530 from different Devices 98 or DCADO Units 100, etc.) can be combined to accumulate collective knowledge of operating Device 98 in circumstances including objects with various properties. In one example, one Knowledgebase 530 can be appended to another Knowledgebase 530 such as appending one Collection of Sequences 530c (later described) to another Collection of Sequences 530c, appending one Sequence 533 (later described) to another Sequence 533, appending one Collection of Knowledge Cells 530d (later described) to another Collection of Knowledge Cells 530d, and/or appending other data structures or elements thereof. In another example, one Knowledgebase 530 can be copied

into another Knowledgebase 530 such as copying one Collection of Sequences 530c into another Collection of Sequences 530c, copying one Collection of Knowledge Cells 530d into another Collection of Knowledge Cells 530d, and/or copying other data structures or elements thereof. In a further example, in the case of Knowledgebase 530 being or including Graph 530b or graph-like data structure (i.e. Neural Network 530a, tree, etc.), a union can be utilized to combine two or more Graphs 530b or graph-like data structures. For instance, a union of two Graphs 530b or graph-like data structures may include a union of their vertex (i.e. node, etc.) sets and their edge (i.e. connection, etc.) sets. Any other operations or combination thereof on graphs or graph-like data structures can be utilized to combine Graphs 530b or graph-like data structures. In a further example, one Knowledgebase 530 can be combined with another Knowledgebase 530 through later described learning processes where Knowledge Cells 800 may be applied one at a time and connected with prior and/or subsequent Knowledge Cells 800 such as in Graph 530b or Neural Network 530a. In such embodiments, instead of Knowledge Cells 800 generated by Knowledge Structuring Unit 520, the learning process may utilize Knowledge Cells 800 from one Knowledgebase 530 to apply them onto another Knowledgebase 530. Any other techniques known in art including custom techniques for combining data structures can be utilized for combining Knowledgebases 530 in alternate implementations. In any of the aforementioned and/or other combining techniques, similarity of elements (i.e. nodes/vertices, edges/connections, etc.) can be utilized in determining whether an element from one Knowledgebase 530 matches an element from another Knowledgebase 530, and substantially or otherwise similar elements may be considered a match for combining purposes in some designs. Any features, functionalities, and embodiments of Similarity Comparison 125 (later described) can be used in such similarity determinations. A combined Knowledgebase 530 can be offered as a network service (i.e. online application, etc.), downloadable file, or other repository to all DCADO Units 100 configured to utilize the combined Knowledgebase 530. For example, a Device 98 including or interfaced with DCADO Unit 100 having access to a combined Knowledgebase 530 can use the collective knowledge learned from multiple Devices 98 for the Device's 98 autonomous operation.

Referring to Fig. 17, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation may include various artificial intelligence models and/or techniques. The disclosed systems, devices, and methods are independent of the artificial intelligence model and/or technique used and any model and/or technique can be utilized to facilitate the functionalities described herein. Examples of these models and/or techniques include deep learning, supervised learning, unsupervised learning, neural networks (i.e. convolutional neural network, recurrent neural network, deep neural network, etc.), search-based, logic and/or fuzzy logic-based, optimization-based, tree/graph/other data structure-based, hierarchical, symbolic and/or sub-symbolic, evolutionary, genetic, multi-agent, deterministic, probabilistic, statistical, and/or other models and/or techniques.

In one example shown in Model A, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation may include a neural network (also referred to as artificial neural network, etc.). As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include a network of Nodes 852 (also referred to as neurons, etc.) and Connections 853 similar to that of a brain. Node 852 can store any data, object, data structure, and/or other item, or reference thereto. Node 852 may also include a function for transforming or manipulating any data, object, data structure, and/or other item. Examples of such transformation

functions include mathematical functions (i.e. addition, subtraction, multiplication, division, sin, cos, log, derivative, integral, etc.), object manipulation functions (i.e. creating an object, modifying an object, deleting an object, appending objects, etc.), data structure manipulation functions (i.e. creating a data structure, modifying a data structure, deleting a data structure, creating a data field, modifying a data field, deleting a data field, etc.), and/or other transformation functions. Connection 853 may include or be associated with a value such as a symbolic label or numeric attribute (i.e. weight, cost, capacity, length, etc.). A computational model can be utilized to compute values from inputs based on a pre-programmed or learned function or method. For example, a neural network may include one or more input neurons that can be activated by inputs. Activations of these neurons can then be passed on, weighted, and transformed by a function to other neurons. Neural networks may range from those with only one layer of single direction logic to multi-layer of multi-directional feedback loops. A neural network can use weights to change the parameters of the network's throughput. A neural network can learn by input from its environment or from self-teaching using written-in rules. A neural network can be utilized as a predictive modeling approach in machine learning. An exemplary embodiment of a neural network (i.e. Neural Network 530a, etc.) is described later.

In another example shown in Model B, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation may include a graph or graph-like data structure. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include Nodes 852 (also referred to as vertices or points, etc.) and Connections 853 (also referred to as edges, arrows, lines, arcs, etc.) organized as a graph. In general, any Node 852 in a graph can be connected to any other Node 852. A Connection 853 may include unordered pair of Nodes 852 in an undirected graph or ordered pair of Nodes 852 in a directed graph. Nodes 852 can be part of the graph structure or external entities represented by indices or references. A graph can be utilized as a predictive modeling approach in machine learning. Nodes 852, Connections 853, and/or other elements or operations of a graph may include any features, functionalities, and embodiments of the aforementioned Nodes 852, Connections 853, and/or other elements or operations of a neural network, and vice versa. An exemplary embodiment of a graph (i.e. Graph 530b, etc.) is described later.

In a further example shown in Model C, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation may include a tree or tree-like data structure. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include Nodes 852 and Connections 853 (also referred to as references, edges, etc.) organized as a tree. In general, a Node 852 in a tree can be connected to any number (i.e. including zero, etc.) of children Nodes 852. A tree can be utilized as a predictive modeling approach in machine learning. Nodes 852, Connections 853, and/or other elements or operations of a tree may include any features, functionalities, and embodiments of the aforementioned Nodes 852, Connections 853, and/or other elements or operations of a neural network and/or graph, and vice versa.

In a further example shown in Model D, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation may include a sequence or sequence-like data structure. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include a structure of Nodes 852 and/or Connections 853 organized as a sequence. In some aspects, Connections 853 may be optionally omitted from a

sequence as the sequential order of Nodes 852 in a sequence may be implied in the structure. A sequence can be utilized as a predictive modeling approach in machine learning. Nodes 852, Connections 853, and/or other elements or operations of a sequence may include any features, functionalities, and embodiments of the aforementioned Nodes 852, Connections 853, and/or other elements or operations of a neural network, graph, and/or tree, and vice versa. An exemplary embodiment of a sequence (i.e. Collection of Sequences 530c, Sequence 533, etc.) is described later.

In yet another example, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation may include a search-based model and/or technique. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include searching through a collection of possible solutions. For example, a search method can search through a neural network, graph, tree, sequence, or other data structure that includes data elements of interest. A search may use heuristics to limit the search for solutions by eliminating choices that are unlikely to lead to the goal. Heuristic techniques may provide a best guess solution. A search can also include optimization. For example, a search may begin with a guess and then refine the guess incrementally until no more refinements can be made. In a further example, the disclosed systems, devices, and methods may include logic-based model and/or technique. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities can use formal or other type of logic. Logic based models may involve making inferences or deriving conclusions from a set of premises. As such, a logic based system can extend existing knowledge or create new knowledge automatically using inferences. Examples of the types of logic that can be utilized include propositional or sentential logic that comprises logic of statements which can be true or false; first-order logic that allows the use of quantifiers and predicates and that can express facts about objects, their properties, and their relations with each other; fuzzy logic that allows degrees of truth to be represented as a value between 0 and 1 rather than simply 0 (false) or 1 (true), which can be used for uncertain reasoning; subjective logic that comprises a type of probabilistic logic that may take uncertainty and belief into account, which can be suitable for modeling and analyzing situations involving uncertainty, incomplete knowledge and different world views; and/or other types of logic. In a further example, the disclosed systems, devices, and methods may include a probabilistic model and/or technique. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities can be implemented to operate with incomplete or uncertain information where probabilities may affect outcomes. Bayesian network, among other models, is an example of a probabilistic tool used for purposes such as reasoning, learning, planning, perception, and/or others. One of ordinary skill in art will understand that the aforementioned artificial intelligence models and/or techniques are described merely as examples of a variety of possible implementations, and that while all possible artificial intelligence models and/or techniques are too voluminous to describe, other artificial intelligence models and/or techniques known in art are within the scope of this disclosure. One of ordinary skill in art will also recognize that an intelligent system may solve a specific problem by using any model and/or technique that works such as, for example, some systems can be symbolic and logical, some can be sub-symbolic neural networks, some can be deterministic or probabilistic, some can be hierarchical, some may include searching techniques, some may include optimization techniques, while

others may use other or a combination of models and/or techniques. In general, any artificial intelligence model and/or technique can be utilized that can facilitate the functionalities described herein.

Referring to Figs. 18A-18C, embodiments of interconnected Knowledge Cells 800 and updating weights of Connections 853 are illustrated. As shown for example in Fig. 18A, Knowledge Cell 800za is connected to Knowledge Cell 800zb and Knowledge Cell 800zc by Connection 853z1 and Connection 853z2, respectively. Each of Connection 853z1 and Connection 853z2 may include or be associated with occurrence count, weight, and/or other parameter or data. The number of occurrences may track or store the number of observations that a Knowledge Cell 800 was followed by another Knowledge Cell 800 indicating a connection or relationship between them. For example, Knowledge Cell 800za was followed by Knowledge Cell 800zb 10 times as indicated by the number of occurrences of Connection 853z1. Also, Knowledge Cell 800za was followed by Knowledge Cell 800zc 15 times as indicated by the number of occurrences of Connection 853z2. The weight of Connection 853z1 can be calculated or determined as the number of occurrences of Connection 853z1 divided by the sum of occurrences of all connections (i.e. Connection 853z1 and Connection 853z2, etc.) originating from Knowledge Cell 800za. Therefore, the weight of Connection 853z1 can be calculated or determined as 10/(10+15)=0.4, for example. Also, the weight of Connection 853z2 can be calculated or determined as 15/(10+15)=0.6, for example. Therefore, the sum of weights of Connection 853z1, Connection 853z2, and/or any other Connections 853 originating from Knowledge Cell 800za may equal to 1 or 100%. As shown for example in Fig. 18B, in the case that Knowledge Cell 800zd is inserted and an observation is made that Knowledge Cell 800zd follows Knowledge Cell 800za, Connection 853z3 can be created between Knowledge Cell 800za and Knowledge Cell 800zd. The occurrence count of Connection 853z3 can be set to 1 and weight determined as 1/(10+15+1)=0.038. The weights of all other connections (i.e. Connection 853z1, Connection 853z2, etc.) originating from Knowledge Cell 800za may be updated to account for the creation of Connection 853z3. Therefore, the weight of Connection 853z1 can be updated as 10/(10+15+1)=0.385. The weight of Connection 853z2 can also be updated as 15/(10+15+1)=0.577. As shown for example in Fig. 18C, in the case that an additional occurrence of Connection 853z1 is observed (i.e. Knowledge Cell 800zb followed Knowledge Cell 800za, etc.), occurrence count of Connection 853z1 and weights of all connections (i.e. Connection 853z1, Connection 853z2, and Connection 853z3, etc.) originating from Knowledge Cell 800za may be updated to account for this observation. The occurrence count of Connection 853z1 can be increased by 1 and its weight updated as 11/(11+15+1)=0.407. The weight of Connection 853z2 can also be updated as 15/(11+15+1)=0.556. The weight of Connection 853z3 can also be updated as 1/(11+15+1)=0.037.

Referring to Fig. 19, an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Collection of Knowledge Cells 530d is illustrated. Collection of Knowledge Cells 530d comprises the functionality for storing any number of Knowledge Cells 800. In some aspects, Knowledge Cells 800 may be stored into or applied onto Collection of Knowledge Cells 530d in a learning or training process. In effect, Collection of Knowledge Cells 530d may store Knowledge Cells 800 that can later be used to enable autonomous Device 98 operation. In some embodiments, Knowledge Structuring Unit 520 structures or generates Knowledge Cells 800 as previously described and the system applies them onto Collection of Knowledge Cells 530d, thereby implementing learning Device's 98 operation in circumstances including objects with various properties. The term apply or applying may refer to storing, copying, inserting, updating, or other similar action, therefore, these terms may be used

interchangeably herein depending on context. The system can perform Similarity Comparisons 125 (later described) of a newly structured Knowledge Cell 800 from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Collection of Knowledge Cells 530d. If a substantially similar Knowledge Cell 800 is not found in Collection of Knowledge Cells 530d, the system may insert (i.e. copy, store, etc.) the Knowledge Cell 800 from Knowledge Structuring Unit 520 into Collection of Knowledge Cells 530d, for example. On the other hand, if a substantially similar Knowledge Cell 800 is found in Collection of Knowledge Cells 530d, the system may optionally omit inserting the Knowledge Cell 800 from Knowledge Structuring Unit 520 as inserting a substantially similar Knowledge Cell 800 may not add much or any additional knowledge to the Collection of Knowledge Cells 530d, for example. Also, inserting a substantially similar Knowledge Cell 800 can optionally be omitted to save storage resources and limit the number of Knowledge Cells 800 that may later need to be processed or compared. Any features, functionalities, and embodiments of Similarity Comparison 125, importance index (later described), similarity index (later described), and/or other disclosed elements can be utilized to facilitate determination of substantial or other similarity and whether to insert a newly structured Knowledge Cell 800 into Collection of Knowledge Cells 530d.

For example, the system can perform Similarity Comparisons 125 (later described) of Knowledge Cell 800ba from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Collection of Knowledge Cells 530d. In the case that a substantially similar match is found between Knowledge Cell 800ba and any of the Knowledge Cells 800 in Collection of Knowledge Cells 530d, the system may perform no action. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bb from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Collection of Knowledge Cells 530d. In the case that a substantially similar match is not found, the system may insert a new Knowledge Cell 800 into Collection of Knowledge Cells 530d and copy Knowledge Cell 800bb into the inserted new Knowledge Cell 800. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bc from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Collection of Knowledge Cells 530d. In the case that a substantially similar match is found between Knowledge Cell 800bc and any of the Knowledge Cells 800 in Collection of Knowledge Cells 530d, the system may perform no action. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bd from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Collection of Knowledge Cells 530d. In the case that a substantially similar match is not found, the system may insert a new Knowledge Cell 800 into Collection of Knowledge Cells 530d and copy Knowledge Cell 800bd into the inserted new Knowledge Cell 800. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800be from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Collection of Knowledge Cells 530d. In the case that a substantially similar match is not found, the system may insert a new Knowledge Cell 800 into Collection of Knowledge Cells 530d and copy Knowledge Cell 800be into the inserted new Knowledge Cell 800. Applying any additional Knowledge Cells 800 from Knowledge Structuring Unit 520 onto Collection of Knowledge Cells 530d follows similar logic or process as the above-described.

Referring to Fig. 20, an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Neural Network 530a is illustrated. Neural Network 530a includes a number of neurons or Nodes 852 interconnected by Connections 853 as previously described. Knowledge Cells 800 are shown instead of Nodes 852 to simplify the illustration as Node 852 includes a Knowledge Cell 800, for example. Therefore, Knowledge Cells 800 and Nodes 852 can be used interchangeably herein depending on context. It should be noted that Node 852 may include other elements

and/or functionalities instead of or in addition to Knowledge Cell 800. In some aspects, Knowledge Cells 800 may be stored into or applied onto Neural Network 530a individually or collectively in a learning or training process. In some designs, Neural Network 530a comprises a number of Layers 854 each of which may include one or more Knowledge Cells 800. Knowledge Cells 800 in successive Layers 854 can be connected by Connections 853. Connection 853 may include or be associated with occurrence count, weight, and/or other parameter or data as previously described. Neural Network 530a may include any number of Layers 854 comprising any number of Knowledge Cells 800. In some aspects, Neural Network 530a may store Knowledge Cells 800 interconnected by Connections 853 where following a path through the Neural Network 530a can later be used to enable autonomous Device 98 operation. It should be understood that, in some embodiments, Knowledge Cells 800 in one Layer 854 of Neural Network 530a need not be connected only with Knowledge Cells 800 in a successive Layer 854, but also in any other Layer 854, thereby creating shortcuts (i.e. shortcut Connections 853, etc.) through Neural Network 530a. A Knowledge Cell 800 can also be connected to itself such as, for example, in recurrent neural networks. In general, any Knowledge Cell 800 can be connected with any other Knowledge Cell 800 anywhere else in Neural Network 530a. In further embodiments, back-propagation of any data or information can be implemented. In one example, back-propagation of similarity (i.e. similarity index, etc.) of compared Knowledge Cells 800 in a path through Neural Network 530a can be implemented. In another example, back-propagation of errors can be implemented. Such back-propagations can then be used to adjust occurrence counts and/or weights of Connections 853 for better future predictions, for example. Any other back-propagation can be implemented for other purposes. Any combination of Nodes 852 (i.e. Nodes 852 comprising Knowledge Cells 800, etc.), Connections 853, Layers 854, and/or other elements or techniques can be implemented in alternate embodiments. Neural Network 530a may include any type or form of a neural network known in art such as a feed-forward neural network, a back-propagating neural network, a recurrent neural network, a convolutional neural network, deep neural network, and/or others including a custom neural network.

In some embodiments, Knowledge Structuring Unit 520 structures or generates Knowledge Cells 800 and the system applies them onto Neural Network 530a, thereby implementing learning Device's 98 operation in circumstances including objects with various properties. The system can perform Similarity Comparisons 125 (later described) of a Knowledge Cell 800 from Knowledge Structuring Unit 520 with Knowledge Cells 800 in a Layer 854 of Neural Network 530a. If a substantially similar Knowledge Cell 800 is not found in the Layer 854 of Neural Network 530a, the system may insert (i.e. copy, store, etc.) the Knowledge Cell 800 from Knowledge Structuring Unit 520 into the Layer 854 of Neural Network 530a, and create a Connection 853 to the inserted Knowledge Cell 800 from a Knowledge Cell 800 in a prior Layer 854 including assigning an occurrence count to the new Connection 853, calculating a weight of the new Connection 853, and updating any other Connections 853 originating from the Knowledge Cell 800 in the prior Layer 854. On the other hand, if a substantially similar Knowledge Cell 800 is found in the Layer 854 of Neural Network 530a, the system may update occurrence count and weight of Connection 853 to that Knowledge Cell 800 from a Knowledge Cell 800 in a prior Layer 854, and update any other Connections 853 originating from the Knowledge Cell 800 in the prior Layer 854.

For example, the system can perform Similarity Comparisons 125 (later described) of Knowledge Cell 800ba from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Layer 854a of Neural Network 530a. In the case that a substantially similar match is found between Knowledge Cell 800ba and Knowledge Cell 800ea, the

system may perform no action since Knowledge Cell 800ea is the initial Knowledge Cell 800. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bb from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Layer 854b of Neural Network 530a. In the case that a substantially similar match is found between Knowledge Cell 800bb and Knowledge Cell 800eb, the system may update occurrence count and weight of Connection 853e1 between Knowledge Cell 800ea and Knowledge Cell 800eb, and update weights of other Connections 853 originating from Knowledge Cell 800ea as previously described. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bc from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Layer 854c of Neural Network 530a. In the case that a substantially similar match is not found, the system may insert Knowledge Cell 800ec into Layer 854c and copy Knowledge Cell 800bc into the inserted Knowledge Cell 800ec. The system may also create Connection 853e2 between Knowledge Cell 800eb and Knowledge Cell 800ec with occurrence count of 1 and weight calculated based on the occurrence count as previously described. The system may also update weights of other Connections 853 (one in this example) originating from Knowledge Cell 800eb as previously described. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bd from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Layer 854d of Neural Network 530a. In the case that a substantially similar match is not found, the system may insert Knowledge Cell 800ed into Layer 854d and copy Knowledge Cell 800bd into the inserted Knowledge Cell 800ed. The system may also create Connection 853e3 between Knowledge Cell 800ec and Knowledge Cell 800ed with occurrence count of 1 and weight of 1. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800be from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Layer 854e of Neural Network 530a. In the case that a substantially similar match is not found, the system may insert Knowledge Cell 800ee into Layer 854e and copy Knowledge Cell 800be into the inserted Knowledge Cell 800ee. The system may also create Connection 853e4 between Knowledge Cell 800ed and Knowledge Cell 800ee with occurrence count of 1 and weight of 1. Applying any additional Knowledge Cells 800 from Knowledge Structuring Unit 520 onto Neural Network 530a follows similar logic or process as the above-described.

Referring now to Similarity Comparison 125, Similarity Comparison 125 comprises the functionality for comparing or matching Knowledge Cells 800 or portions thereof, and/or other functionalities. Similarity Comparison 125 comprises the functionality for comparing or matching Collections of Object Representations 525 or portions thereof. Similarity Comparison 125 comprises the functionality for comparing or matching streams of Collections of Object Representations 525 or portions thereof. Similarity Comparison 125 comprises the functionality for comparing or matching Object Representations 625 or portions thereof. Similarity Comparison 125 comprises the functionality for comparing or matching Object Properties 630 or portions thereof. Similarity Comparison 125 comprises the functionality for comparing or matching Instruction Sets 526, Extra Info 527, text (i.e. characters, words, phrases, etc.), numbers, and/or other elements or portions thereof. Similarity Comparison 125 may include functions, rules, and/or logic for performing matching or comparisons and for determining that while a perfect match is not found, a partial or similar match has been found. In some aspects, a partial match may include a substantially or otherwise similar match, and vice versa. Therefore, these terms may be used interchangeably herein depending on context. As such, Similarity Comparison 125 may include determining substantial similarity or substantial match of compared elements. Although, substantial similarity or substantial match is frequently used herein, it should be understood that any level of similarity, however high or low, may be utilized as defined by the rules (i.e. thresholds, etc.) for similarity.

The rules for similarity or similar match can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. In some designs, Similarity Comparison 125 comprises the functionality to automatically define appropriately strict rules for determining similarity of the compared elements. Similarity Comparison 125 can therefore set, reset, and/or adjust the strictness of the rules for finding or determining similarity of the compared elements, thereby fine tuning Similarity Comparison 125 so that the rules for determining similarity are appropriately strict. In some aspects, the rules for determining similarity may include a similarity threshold. As such, Similarity Comparison 125 can determine similarity of compared elements if their similarity exceeds a similarity threshold. In other aspects, the rules for determining similarity may include a difference threshold. As such, Similarity Comparison 125 can determine similarity of compared elements if their difference is lower than a difference threshold. In further aspects, the rules for determining similarity may include other thresholds. Similarity Comparison 125 enables comparing circumstances including objects with various properties and determining their similarity or match. In one example, a circumstance including an object detected at a distance of 8m and an angle/bearing of 64° relative to Device 98 may be found similar or matching by Similarity Comparison 125 to a circumstance including the same or similar object detected at a distance of 8.6m and an angle/bearing of 59° relative to Device 98. In another example, a circumstance including an object detected as a passenger vehicle may be found similar or matching by Similarity Comparison 125 to a circumstance including an object detected as a sport utility vehicle. In general, any one or more properties (i.e. existence, type, identity, distance, bearing/angle, location, shape/size, activity, etc.) of one or more objects can be utilized for determining similarity or match of circumstances including objects with various properties. Therefore, Similarity Comparison 125 provides flexibility in comparing and determining similarity of a variety of possible circumstances of Device 98.

In some embodiments where compared Knowledge Cells 800 include a single Collection of Object Representations 525, in determining similarity of Knowledge Cells 800, Similarity Comparison 125 can perform comparison of individual Collections of Object Representations 525 or portions (i.e. Object Representations 625, Object Properties 630, etc.) thereof such as comparison of Collection of Object Representations 525 or portions thereof from one Knowledge Cell 800 with Collection of Object Representations 525 or portions thereof from another Knowledge Cell 800. In some aspects, total equivalence is achieved when Collection of Object Representations 525 or portions thereof from one Knowledge Cell 800 matches Collection of Object Representations 525 or portions thereof from another Knowledge Cell 800. If total equivalence is not found, Similarity Comparison 125 may attempt to determine substantial or other similarity of compared Knowledge Cells 800.

In some embodiments, in determining substantial similarity of individually compared Collections of Object Representations 525 (i.e. Collections of Object Representations 525 from the compared Knowledge Cells 800, etc.), Similarity Comparison 125 can compare one or more Object Representations 625 or portions (i.e. Object Properties 630, etc.) thereof from one Collection of Object Representations 525 with one or more Object Representations 625 or portions thereof from another Collection of Object Representations 525. In some aspects, total equivalence is found when all Object Representations 625 or portions thereof from one Collection of Object Representations 525 match all Object Representations 625 or portions thereof from another Collection of Object Representations 525. In other aspects, if total equivalence is not found, Similarity Comparison 125 may attempt to determine substantial similarity of compared Collections of Object Representations 525. In one example, substantial similarity can be

achieved when most of the Object Representations 625 or portions thereof from the compared Collections of Object Representations 525 match or substantially match. In another example, substantial similarity can be achieved when at least a threshold number (i.e. 1, 2, 4, 7, 18, etc.) or percentage (i.e. 41%, 62%, 79%, 85%, 93%, etc.) of Object Representations 625 or portions thereof from the compared Collections of Object Representations 525 match or substantially match. Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching Object Representations 625 or portions thereof from the compared Collections of Object Representations 525 exceeds a threshold number (i.e. 1, 2, 4, 7, 18, etc.) or a threshold percentage (i.e. 41%, 62%, 79%, 85%, 93%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or percentage of Object Representations 625 or portions thereof from the compared Collections of Object Representations 525 match or substantially match. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In some aspects, Similarity Comparison 125 can utilize the importance (i.e. as indicated by importance index [later described], etc.) of Object Representations 625 or portions thereof for determining substantial similarity of Collections of Object Representations 525. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to more important Object Representations 625 or portions thereof such as Object Representations 625 representing near Objects 615, Object Representations 625 representing large Objects 615, etc., thereby tolerating mismatches in less important Object Representations 625 or portions thereof such as Object Representations 625 representing distant Objects 615, Object Representations 625 representing small Objects 615, etc. In general, any Object Representation 625 or portion thereof can be assigned higher or lower importance. In further aspects, Similarity Comparison 125 can omit some of the Object Representations 625 or portions thereof from the comparison in determining substantial similarity of Collections of Object Representations 525. In one example, Object Representations 625 representing distant Objects 615 can be omitted from comparison. In another example, Object Representations 625 representing small Objects 615 can be omitted from comparison. In general, any Object Representation 625 or portion thereof can be omitted from comparison depending on implementation.

Similarity Comparison 125 can automatically adjust (i.e. increase or decrease) the strictness of the rules for determining substantial similarity of Collections of Object Representations 525. In some aspects, such adjustment in strictness can be done by Similarity Comparison 125 in response to determining that total equivalence of compared Collections of Object Representations 525 had not been found. Similarity Comparison 125 can keep adjusting the strictness rules until a substantial similarity is found. All the rules or settings of substantial similarity can be set, reset, or adjusted by Similarity Comparison 125 in response to another strictness level determination. For example, Similarity Comparison 125 may attempt to find a match or substantial match in a certain percentage (i.e. 81%, etc.) of Object Representations 625 or portions thereof from the compared Collections of Object Representations 525. If the comparison does not determine substantial similarity of compared Collections of Object Representations 525, Similarity Comparison 125 may decide to decrease the strictness of the rules. In response, Similarity Comparison 125 may attempt to find fewer matching or substantially matching Object Representations 625 or portions thereof than in the previous attempt using stricter rules. If the comparison still does not determine substantial similarity of compared Collections of Object Representations 525, Similarity Comparison 125 may further decrease the strictness (i.e. down to a certain minimum strictness or threshold, etc.) by requiring fewer Object Representations 625 or

portions thereof to match or substantially match, thereby further increasing a chance of finding substantial similarity in compared Collections of Object Representations 525.

Where a reference to Object Representation 625 is used herein it should be understood that a portion of Object Representation 625 (i.e. Object Property 630, etc.) or a plurality of Object Representations 625 can be used instead of or in addition to the Object Representation 625. In one example, instead of or in addition to Object Representation 625, Object Properties 630 and/or other portions that constitute an Object Representation 625 can be compared. In another example, instead of or in addition to Object Representation 625, plurality of Object Representations 625 can be compared. As such, any operations, rules, logic, and/or functions operating on Object Representation 625 may similarly apply to any portion of Object Representation 625 and/or a plurality of Object Representations 625 as applicable. In general, whole Object Representations 625, portions of Object Representations 625, and/or pluralities of Object Representations 625, including any operations thereon, can be combined to arrive at desired results. Some or all of the above-described rules, logic, and/or techniques can be utilized alone or in combination with each other or with other rules, logic, and/or techniques. One of ordinary skill in art will recognize that other techniques known in art for determining similarity of Object Representations 625 and/or other data that would be too voluminous to describe are within the scope of this disclosure.

In some embodiments, in determining substantial similarity of Object Representations 625 (i.e. Object Representations 625 from the compared Collections of Object Representations 525, etc.), Similarity Comparison 125 can compare Object Properties 630 or portions (i.e. characters, words, numbers, etc.) thereof from one Object Representation 625 with Object Properties 630 or portions thereof from another Object Representation 625. In some aspects, total equivalence is found when all Object Properties 630 or portions thereof of one Object Representation 625 match all Object Properties 630 or portions thereof of another Object Representation 625. In other aspects, if total equivalence is not found, Similarity Comparison 125 may attempt to determine substantial similarity of compared Object Representations 625. In one example, substantial similarity can be achieved when most of the Object Properties 630 or portions thereof from the compared Object Representations 625 match or substantially match. In another example, substantial similarity can be achieved when at least a threshold number (i.e. 1, 2, 3, 6, 11, etc.) or percentage (i.e. 55%, 61%, 78%, 82%, 99%, etc.) of Object Properties 630 or portions thereof from the compared Object Representations 625 match or substantially match. Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching Object Properties 630 or portions thereof from the compared Object Representations 625 exceeds a threshold number (i.e. 1, 2, 3, 6, 11, etc.) or a threshold percentage (i.e. 55%, 61%, 78%, 82%, 99%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or percentage of Object Properties 630 or portions thereof from the compared Object Representations 625 match or substantially match. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In further aspects, Similarity Comparison 125 can utilize Categories 635 associated with Object Properties 630 for determining substantial similarity of Object Representations 625. In one example, Object Properties 630 or portions thereof from the compared Object Representations 625 in a same Category 635 may be compared. This way, Object Properties 630 or portions thereof can be compared with their own peers. In one instance, Object Properties 630 or portions thereof from the compared Object Representations 625 in Category 635 "Type" may be compared. Any text comparison technique can be utilized in such comparing. In another

instance, Object Properties 630 or portions thereof from the compared Object Representations 625 in Category 635 "Distance" or "Bearing" may be compared. Any number comparison technique can be utilized in such comparing. In a further instance, Object Properties 630 or portions thereof from the compared Object Representations 625 in Category 635 "Shape" may be compared. Any model, point cloud, or other computer construct comparison technique can be utilized in such comparing. In further aspects, Similarity Comparison 125 can utilize the importance (i.e. as indicated by importance index [later described], etc.) of Object Properties 630 or portions thereof for determining substantial similarity of Object Representations 625. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to more important Object Properties 630 or portions thereof such as Object Properties 630 or portions thereof in Categories 635 "Type", "Distance", "Bearing", etc., thereby tolerating mismatches in less important Object Properties 630 or portions thereof such as Object Properties 630 or portions thereof in Categories 635 "Identity", "Shape", etc. In general, any Object Property 630 or portion thereof can be assigned higher or lower importance. In further aspects, Similarity Comparison 125 can omit some of the Object Properties 630 or portions thereof from the comparison in determining substantial similarity of Object Representations 625. In one example, Object Properties 630 or portions thereof in Category 635 "Identity" can be omitted from comparison. In another example, Object Properties 630 or portions thereof in Category 635 "Shape" can be omitted from comparison. In general, any Object Property 630 or portion thereof can be omitted from comparison depending on implementation.

Similarity Comparison 125 can automatically adjust (i.e. increase or decrease) the strictness of the rules for determining substantial similarity of Object Representations 625. In some aspects, such adjustment in strictness can be done by Similarity Comparison 125 in response to determining that total equivalence of compared Object Representations 625 had not been found. Similarity Comparison 125 can keep adjusting the strictness rules until a substantial similarity is found. All the rules or settings of substantial similarity can be set, reset, or adjusted by Similarity Comparison 125 in response to another strictness level determination. For example, Similarity Comparison 125 may attempt to find a match or substantial match in a certain percentage (i.e. 87%, etc.) of Object Properties 630 or portions thereof from the compared Object Representations 625. If the comparison does not determine substantial similarity of compared Object Representations 625, Similarity Comparison 125 may decide to decrease the strictness of the rules. In response, Similarity Comparison 125 may attempt to find fewer matching or substantially matching Object Properties 630 or portions thereof than in the previous attempt using stricter rules. If the comparison still does not determine substantial similarity of compared Object Representations 625, Similarity Comparison 125 may further decrease the strictness (i.e. down to a certain minimum strictness or threshold, etc.) by requiring fewer Object Properties 630 or portions thereof to match or substantially match, thereby further increasing a chance of finding substantial similarity in compared Object Representations 625. In further aspects, an adjustment in strictness can be done by Similarity Comparison 125 in response to determining that multiple substantially similar Object Representations 625 had been found. Similarity Comparison 125 can keep adjusting the strictness of the rules until a best of the substantially similar Object Representations 625 is found. For example, Similarity Comparison 125 may attempt to find a match or substantial match in a certain percentage (i.e. 65%, etc.) of Object Properties 630 or portions thereof from the compared Object Representations 625. If the comparison determines a number of substantially similar Object Representations 625, Similarity Comparison 125 may decide to increase the strictness of the rules to decrease the number of substantially similar Object Representations 625. In response,

Similarity Comparison 125 may attempt to find more matching or substantially matching Object Properties 630 or portions thereof in addition to the earlier found Object Properties 630 or portions thereof to limit the number of substantially similar Object Representations 625. If the comparison still provides more than one substantially similar Object Representation 625, Similarity Comparison 125 may further increase the strictness by requiring additional Object Properties 630 or portions thereof to match or substantially match, thereby further narrowing the number of substantially similar Object Representations 625 until a best substantially similar Object Representation 625 is found.

Where a reference to Object Property 630 is used herein it should be understood that a portion of Object Property 630 or a plurality of Object Properties 630 can be used instead of or in addition to the Object Property 630. In one example, instead of or in addition to Object Property 630, characters, words, numbers, and/or other portions that constitute an Object Property 630 can be compared. In another example, instead of or in addition to Object Property 630, a plurality of Object Properties 630 can be compared. As such, any operations, rules, logic, and/or functions operating on Object Property 630 may similarly apply to any portion of Object Property 630 and/or a plurality of Object Properties 630 as applicable. In general, whole Object Properties 630, portions of Object Properties 630, and/or pluralities of Object Properties 630, including any operations thereon, can be combined to arrive at desired results. Some or all of the above-described rules, logic, and/or techniques can be utilized alone or in combination with each other or with other rules, logic, and/or techniques. One of ordinary skill in art will recognize that other techniques known in art for determining similarity of Object Properties 630 and/or other data that would be too voluminous to describe are within the scope of this disclosure.

In some embodiments where compared Knowledge Cells 800 include a stream of Collections of Object Representations 525, in determining similarity of Knowledge Cells 800, Similarity Comparison 125 can perform collective comparison of Collections of Object Representations 525 or portions (i.e. Object Representations 625, Object Properties 630, etc.) thereof such as comparison of a stream of Collections of Object Representations 525 or portions thereof from one Knowledge Cell 800 with a stream of Collections of Object Representations 525 or portions thereof from another Knowledge Cell 800. Similarity Comparison 125 of collectively compared Collections of Object Representations 525 or portions thereof may include any features, functionalities, and embodiments of the previously described Similarity Comparison 125 of individually compared Collections of Object Representations 525 or portions thereof. In some aspects, total equivalence is found when all Collections of Object Representations 525 or portions thereof from one Knowledge Cell 800 match all Collections of Object Representations 525 or portions thereof from another Knowledge Cell 800. If total equivalence is not found, Similarity Comparison 125 may attempt to determine substantial or other similarity of compared Knowledge Cells 800. In one example, substantial similarity can be achieved when most of the Collections of Object Representations 525 or portions thereof from the compared Knowledge Cells 800 match or substantially match. In another example, substantial similarity can be achieved when at least a threshold number (i.e. 1, 2, 4, 9, 33, 138, etc.) or percentage (i.e. 39%, 58%, 77%, 88%, 94%, etc.) of Collections of Object Representations 525 or portions thereof from the compared Knowledge Cells 800 match or substantially match. Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching Collections of Object Representations 525 or portions thereof from the compared Knowledge Cells 800 exceeds a threshold number (i.e. 1, 2, 4, 9, 33, 138, etc.) or a threshold percentage (i.e. 39%, 58%, 77%, 88%, 94%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or

percentage of Collections of Object Representations 525 or portions thereof from the compared Knowledge Cells 800 match or substantially match. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In some aspects, Similarity Comparison 125 can utilize the importance (i.e. as indicated by importance index [later described], etc.) of Collections of Object Representations 525 or portions thereof for determining substantial similarity of Knowledge Cells 800. In one example, substantial similarity can be achieved when matches or substantial matches are found with respect to more important Collections of Object Representations 525 or portions thereof such as more substantive or larger Collections of Object Representations 525 (i.e. Collections of Object Representations 525 comprising a higher number of Object Representations 625, etc.) or portions thereof, etc., thereby tolerating mismatches in less important Collections of Object Representations 525 or portions thereof such as less substantive or smaller Collections of Object Representations 525 (i.e. Collections of Object Representations 525 comprising a lower number of Object Representations 625, etc.) or portions thereof, etc. In general, any Collection of Object Representations 525 or portion thereof can be assigned higher or lower importance. In other aspects, Similarity Comparison 125 can utilize the order of Collections of Object Representations 525 or portions thereof for determining substantial similarity of Knowledge Cells 800. In one example, substantial similarity can be achieved when matches or substantial matches are found in earlier Collections of Object Representations 525 or portions thereof from the compared Knowledge Cells 800, thereby tolerating mismatches in later Collections of Object Representations 525 or portions thereof. In another example, substantial similarity can be achieved when matches or substantial matches are found in corresponding (i.e. similarly ordered, temporally related, etc.) Collections of Object Representations 525 or portions thereof from the compared Knowledge Cells 800. In one instance, a 94th Collection of Object Representations 525 or portions thereof from one Knowledge Cell 800 can be compared with a 94th Collection of Object Representations 525 or portions thereof from another Knowledge Cell 800. In another instance, a 94th Collection of Object Representations 525 or portions thereof from one Knowledge Cell 800 can be compared with a number of Collections of Object Representations 525 or portions thereof around (i.e. preceding and/or following) a 94th Collection of Object Representations 525 from another Knowledge Cell 800. This way, flexibility can be implemented in finding a substantially similar Collection of Object Representations 525 or portions thereof if the Collections of Object Representations 525 or portions thereof in the compared Knowledge Cells 800 are not perfectly aligned. In a further instance, Similarity Comparison 125 can utilize Dynamic Time Warping (DTW) and/or other techniques known in art for comparing and/or aligning temporal sequences (i.e. streams of Collections of Object Representations 525 or portions thereof, etc.) that may vary in time or speed. In further aspects, Similarity Comparison 125 can omit some of the Collections of Object Representations 525 or portions thereof from the comparison in determining substantial similarity of Knowledge Cells 800. In one example, less substantive or smaller Collections of Object Representations 525 or portions thereof can be omitted from comparison. In another example, some or all Collections of Object Representations 525 or portions thereof related to a specific time period can be omitted from comparison. In general, any Collection of Object Representations 525 or portion thereof can be omitted from comparison depending on implementation.

Similarity Comparison 125 can automatically adjust (i.e. increase or decrease) the strictness of the rules for determining substantial similarity of Knowledge Cells 800. In some aspects, such adjustment in strictness can be done by Similarity Comparison 125 in response to determining that total equivalence of compared Knowledge Cells

800 had not been found. Similarity Comparison 125 can keep adjusting the strictness of the rules until substantial similarity is found. All the rules or settings of substantial similarity can be set, reset, or adjusted by Similarity Comparison 125 in response to another strictness level determination. For example, Similarity Comparison 125 may attempt to find a match or substantial match in a certain percentage (i.e. 92%, etc.) of Collections of Object Representations 525 or portions thereof from the compared Knowledge Cells 800. If the comparison does not determine substantial similarity of compared Knowledge Cells 800, Similarity Comparison 125 may decide to decrease the strictness of the rules. In response, Similarity Comparison 125 may attempt to find fewer matching or substantially matching Collections of Object Representations 525 or portions thereof than in the previous attempt using stricter rules. If the comparison still does not determine substantial similarity of compared Knowledge Cells 800, Similarity Comparison 125 may further decrease (i.e. down to a certain minimum strictness or threshold, etc.) the strictness by requiring fewer Collections of Object Representations 525 or portions thereof to match or substantially match, thereby further increasing a chance of finding substantial similarity in compared Knowledge Cells 800. In further aspects, an adjustment in strictness can be done by Similarity Comparison 125 in response to determining that multiple substantially similar Knowledge Cells 800 had been found. Similarity Comparison 125 can keep adjusting the strictness of the rules until a best of the substantially similar Knowledge Cells 800 is found. For example, Similarity Comparison 125 may attempt to find a match or substantial match in a certain percentage (i.e. 71%, etc.) of Collections of Object Representations 525 or portions thereof from the compared Knowledge Cells 800. If the comparison determines a number of substantially similar Knowledge Cells 800, Similarity Comparison 125 may decide to increase the strictness of the rules to decrease the number of substantially similar Knowledge Cells 800. In response, Similarity Comparison 125 may attempt to find more matching or substantially matching Collections of Object Representations 525 or portions thereof in addition to the earlier found Collections of Object Representations 525 or portions thereof to limit the number of substantially similar Knowledge Cells 800. If the comparison still provides more than one substantially similar Knowledge Cell 800, Similarity Comparison 125 may further increase the strictness by requiring additional Collections of Object Representations 525 or portions thereof to match or substantially match, thereby further narrowing the number of substantially similar Knowledge Cells 800 until a best substantially similar Knowledge Cell 800 is found.

Some or all of the aforementioned rules, logic, and/or techniques for determining substantial similarity of Knowledge Cells 800 can be utilized alone or in combination with each other or with other rules, logic, and/or techniques. One of ordinary skill in art will recognize that other techniques known in art for determining similarity of Knowledge Cells 800 and/or other data that would be too voluminous to describe are within the scope of this disclosure.

In any of the comparisons involving numbers such as, for example, Object Properties 630 including numbers (i.e. distances, bearings/angles, etc.), Similarity Comparison 125 can compare a number from one Object Property 630 with a number from another Object Property 630. In some aspects, total equivalence is found when the number from one Object Property 630 equals the number from another Object Property 630. In other aspects, if total equality is not found, Similarity Comparison 125 may attempt to determine substantial similarity of the compared numbers using a tolerance or threshold for determining a match. In some aspects, Similarity Comparison 125 can utilize a threshold for acceptable number difference in determining a match of compared numbers. For example, a threshold for acceptable number difference (i.e. absolute difference, etc.) can be set at 10. Therefore, 130 matches

or is sufficiently similar to 135 because the number difference (i.e. 5 in this example) is lower than the threshold for acceptable number difference (i.e. 10 in this example, etc.). Furthermore, 130 does not match or is not sufficiently similar to 143 because the number difference (i.e. 13 in this example) is greater than the threshold for acceptable number difference. Any other threshold for acceptable number difference can be used such as 0.024, 1, 8, 15, 77, 197, 2438, 728322, and/or others. In other aspects, Similarity Comparison 125 can utilize a threshold for acceptable percentage difference in determining a match of compared numbers. For example, a threshold for acceptable percentage difference can be set at 10%. Therefore, 100 matches or is sufficiently similar to 106 because the percentage difference (i.e. 6% in this example) is lower than the threshold for acceptable percentage difference (i.e. 10% in this example). Furthermore, 100 does not match or is not sufficiently similar to 84 because the percentage difference (i.e. 16% in this example) is higher than the threshold for acceptable percentage difference. Any other threshold for acceptable percentage difference can be used such as 0.68%, 1%, 3%, 11%, 33%, 69%, 87%, and/or others. The aforementioned thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. Other techniques known in art for comparing numbers can be utilized herein.

In any of the comparisons involving text such as, for example, Object Properties 630 including text (i.e. types, identities, etc.), Similarity Comparison 125 can compare words, characters, and/or other text from one Object Property 630 with words, characters, and/or other text from another Object Property 630. In some aspects, total equivalence is found when all words, characters, and/or other text from one Object Property 630 match all words, characters, and/or other text from another Object Property 630. In other aspects, if total equivalence is not found, Similarity Comparison 125 may attempt to determine substantial similarity of compared Object Properties 630. In one example, substantial similarity can be achieved when most of the words, characters, and/or other text from the compared Object Properties 630 match or substantially match. In another example, substantial similarity can be achieved when at least a threshold number (i.e. 1, 2, 3, 4, 7, 11, etc.) or percentage (i.e. 38%, 63%, 77%, 84%, 98%, etc.) of words, characters, and/or other text from the compared Object Properties 630 match or substantially match. Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching words, characters, and/or other text from the compared Object Properties 630 exceeds a threshold number (i.e. 1, 2, 3, 4, 7, 11, etc.) or a threshold percentage (i.e. 48%, 63%, 77%, 84%, 98%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or percentage of words, characters, and/or other text from the compared Object Properties 630 match or substantially match. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In further aspects, Similarity Comparison 125 can utilize the importance (i.e. as indicated by importance index [later described], etc.) of words, characters, and/or other text for determining substantial similarity of Object Properties 630. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to more important words, characters, and/or other text such as longer words and/or other text, thereby tolerating mismatches in less important words, characters, and/or other text such as shorter words and/or other text. In general, any word, character, and/or other text can be assigned higher or lower importance. In further aspects, Similarity Comparison 125 can utilize the order of words, characters, and/or other text for determining substantial similarity of Object Properties 630. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to front-most

words, characters, and/or other text, thereby tolerating mismatches in later words, characters, and/or other text. In further aspects, Similarity Comparison 125 can utilize semantic conversion to account for variations of words and/or other text. In one example, Object Property 630 may include a word "house". In addition to searching for the exact word in a compared Object Property 630, Similarity Comparison 125 can employ semantic conversion and attempt to match "home", "residence", "dwelling", "place", or other semantically similar variations of the word with a meaning "house". In another example, Object Property 630 may include a word "buy". In addition to searching for the exact word in a compared Object Property 630, Similarity Comparison 125 can employ semantic conversion and attempt to match "buying", "bought", or other semantically similar variations of the word with a meaning "buy" in different tenses. Any other grammatical analysis or transformation can be utilized to cover the full scope of word and/or other text variations. In some designs, semantic conversion can be implemented using a thesaurus or dictionary. In another example, semantic conversion can be implemented using a table where each row comprises semantically similar variations of a word and/or other text. In further aspects, Similarity Comparison 125 can utilize a language model for understanding or interpreting the concepts contained in the words and/or other text and compare the concepts instead of or in addition to the words and/or other text. Examples of language models include unigram model, n-gram model, neural network language model, bag of words model, and/or others. Any of the techniques for matching of words can similarly be used for matching of concepts. In further aspects, Similarity Comparison 125 can omit some of the words, characters, and/or other text from the comparison in determining substantial similarity of Object Properties 630. In one example, rear-most words, characters, and/or other text can be omitted from comparison. In another example, shorter words and/or other text can be omitted from comparison. In general, any word, character, and/or other text can be omitted from comparison depending on implementation. Other techniques known in art for comparing words, characters, and/or other text can be utilized herein.

In some embodiments, Similarity Comparison 125 can compare one or more Extra Info 527 (i.e. time information, location information, computed information, contextual information, and/or other information, etc.) in addition to or instead of comparing Collections of Object Representations 525 or portions thereof in determining substantial similarity of Knowledge Cells 800. Extra Info 527 can be set to be less, equally, or more important (i.e. as indicated by importance index [later described], etc.) than Collections of Object Representations 525, Object Representations 625, Object Properties 630, and/or other elements in the comparison. Since Extra Info 527 may include any contextual or other information that can be useful in determining similarity of any compared elements, Extra Info 527 can be used to enhance any of the aforementioned similarity determinations as applicable.

In some embodiments, Similarity Comparison 125 can also compare one or more Instruction Sets 526 in addition to or instead of comparing Collections of Object Representations 525 or portions thereof in determining substantial similarity of Knowledge Cells 800. In some aspects, Similarity Comparison 125 can compare portions of Instruction Sets 526 to determine substantial or other similarity of Instruction Sets 526. Similar to the above-described thresholds, thresholds for the number or percentage of matching portions of the compared Instruction Sets 526 can be utilized in determining substantial or other similarity of the compared Instruction Sets 526. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In other aspects, Similarity Comparison 125 can compare text (i.e. characters, words, phrases, etc.), numbers, or other data (i.e. bits, etc.) to determine substantial or other similarity of Instruction Sets 526. Any other comparison technique can be

utilized in comparing Instruction Sets 526 in alternate implementations. Instruction Sets 526 can be set to be less, equally, or more important (i.e. as indicated by importance index [later described], etc.) than Collections of Object Representations 525, Object Representations 625, Object Properties 630, Extra Info 527, and/or other elements in the comparison.

In some embodiments, an importance index (not shown) or other importance ranking technique can be used in any of the previously described comparisons or other processing involving elements of different importance. Importance index indicates importance of the element to or with which the index is assigned or associated. For example, importance index may indicate importance of a Knowledge Cell 800, Collection of Object Representations 525, Object Representation 625, Object Property 630, Instruction Set 526, Extra Info 527, and/or other element to or with which the index is assigned or associated. In some aspects, importance index on a scale from 0 to 1 can be utilized, although, any other range can also be utilized. Importance index can be stored in or associated with the element to which the index pertains. Importance indexes of various elements can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. In one example, a higher Importance index can be assigned to more substantive or larger Collections of Object Representations 525 (i.e. Collections of Object Representations 525 comprising a higher number of Object Representations 625, etc.). In another example, a higher importance index can be assigned to Object Representations 625 representing closer, larger, and/or other Objects 615. Any importance index can be assigned to or associated with any element described herein depending on implementation. Any importance ranking technique can be utilized as or instead of importance index in alternate embodiments.

In some embodiments, Similarity Comparison 125 may generate a similarity index (not shown) for any of the compared elements. Similarity index indicates how well an element is matched with another element. For example, similarity index indicates how well a Knowledge Cell 800, Collection of Object Representations 525, Object Representation 625, Object Property 630, Instruction Set 526, Extra Info 527, and/or other element is matched with a compared element. In some aspects, similarity index on a scale from 0 to 1 can be utilized, although, any other range can also be utilized. Similarity index can be generated by Similarity Comparison 125 whether substantial or other similarity between the compared elements is achieved or not. In one example, similarity index can be determined for a Knowledge Cell 800 based on a ratio/percentage of matched or substantially matched Collections of Object Representations 525 relative to the number of Collections of Object Representations 525 in the compared Knowledge Cell 800. Specifically, similarity index of 0.91 is determined if 91% of Collections of Object Representations 525 of one Knowledge Cell 800 match or substantially match Collections of Object Representations 525 of another Knowledge Cell 800. In some designs, importance (i.e. as indicated by importance index, etc.) of one or more Collections of Object Representations 525 can be included in the calculation of a weighted similarity index. Similar determination of similarity index can be implemented with Collections of Object Representations 525, Object Representations 625, Object Properties 630, Instruction Sets 526, Extra Info 527, and/or other elements or portions thereof. Any combination of the aforementioned similarity index determinations or calculations can be utilized in alternate embodiments. Any similarity ranking technique can be utilized to determine or calculate similarity index in alternate embodiments.

Referring to Fig. 21, an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Neural Network 530a comprising shortcut Connections 853 is illustrated. In some designs, Knowledge Cells 800 in one Layer 854 of Neural Network 530a can be connected with Knowledge Cells 800 in any Layer 854, not only in a successive Layer 854, thereby creating shortcuts (i.e. shortcut Connections 853, etc.) through Neural Network 530a. In some aspects, creating a shortcut Connection 853 can be implemented by performing Similarity Comparisons 125 of a Knowledge Cell 800 from Knowledge Structuring Unit 520 with Knowledge Cells 800 in any Layer 854 when applying (i.e. storing, copying, etc.) the Knowledge Cell 800 from Knowledge Structuring Unit 520 onto Neural Network 530a. Once created, shortcut Connections 853 enable a wider variety of Knowledge Cells 800 to be considered when selecting a path through Neural Network 530a. In some embodiments, Knowledge Structuring Unit 520 structures or generates Knowledge Cells 800 and the system applies them onto Neural Network 530a, thereby implementing learning Device's 98 operation in circumstances including objects with various properties. The system can perform Similarity Comparisons 125 of a Knowledge Cell 800 from Knowledge Structuring Unit 520 with Knowledge Cells 800 in one or more Layers 854 of Neural Network 530a. If a substantially similar Knowledge Cell 800 is not found in the one or more Layers 854 of Neural Network 530a, the system may insert (i.e. copy, store, etc.) the Knowledge Cell 800 from Knowledge Structuring Unit 520 into a Layer 854 of Neural Network 530a, and create a Connection 853 to the inserted Knowledge Cell 800 from a prior Knowledge Cell 800 including assigning an occurrence count to the new Connection 853, calculating a weight of the new Connection 853, and updating any other Connections 853 originating from the prior Knowledge Cell 800. On the other hand, if a substantially similar Knowledge Cell 800 is found in the one or more Layers 854 of Neural Network 530a, the system may update occurrence count and weight of Connection 853 to that Knowledge Cell 800 from a prior Knowledge Cell 800, and update any other Connections 853 originating from the prior Knowledge Cell 800. Any of the previously described and/or other techniques for comparing, inserting, updating, and/or other operations on Knowledge Cells 800, Connections 853, Layers 854, and/or other elements can similarly be utilized in Neural Network 530a that comprises shortcut Connections 853.

Referring to Fig. 22, an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Graph 530b is illustrated. In some aspects, any Knowledge Cell 800 can be connected with any other Knowledge Cell 800 in Graph 530b. In other aspects, any Knowledge Cell 800 can be connected with itself and/or any other Knowledge Cell 800 in Graph 530b. In some embodiments, Knowledge Structuring Unit 520 structures or generates Knowledge Cells 800 and the system applies (i.e. store, copy, etc.) them onto Graph 530b, thereby implementing learning Device's 98 operation in circumstances including objects with various properties. The system can perform Similarity Comparisons 125 of a Knowledge Cell 800 from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Graph 530b. If a substantially similar Knowledge Cell 800 is not found in Graph 530b, the system may insert (i.e. copy, store, etc.) the Knowledge Cell 800 from Knowledge Structuring Unit 520 into Graph 530b, and create a Connection 853 to the inserted Knowledge Cell 800 from a prior Knowledge Cell 800 including assigning an occurrence count to the new Connection 853, calculating a weight of the new Connection 853, and updating any other Connections 853 originating from the prior Knowledge Cell 800. On the other hand, if a substantially similar Knowledge Cell 800 is found in Graph 530b, the system may update occurrence count and weight of Connection 853 to that Knowledge Cell 800 from a prior Knowledge Cell 800, and update any other Connections 853 originating from the prior

Knowledge Cell 800. Any of the previously described and/or other techniques for comparing, inserting, updating, and/or other operations on Knowledge Cells 800, Connections 853, and/or other elements can similarly be utilized in Graph 530b.

For example, the system can perform Similarity Comparisons 125 of Knowledge Cell 800ba from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Graph 530b. In the case that a substantially similar match is not found, the system may insert Knowledge Cell 800ha into Graph 530b and copy Knowledge Cell 800ba into the inserted Knowledge Cell 800ha. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bb from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Graph 530b. In the case that a substantially similar match is found between Knowledge Cell 800bb and Knowledge Cell 800hb, the system may create Connection 853h1 between Knowledge Cell 800ha and Knowledge Cell 800hb with occurrence count of 1 and weight of 1. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bc from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Graph 530b. In the case that a substantially similar match is found between Knowledge Cell 800bc and Knowledge Cell 800hc, the system may update occurrence count and weight of Connection 853h2 between Knowledge Cell 800hb and Knowledge Cell 800hc, and update weights of other outgoing Connections 853 (one in this example) originating from Knowledge Cell 800hb as previously described. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bd from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Graph 530b. In the case that a substantially similar match is not found, the system may insert Knowledge Cell 800hd into Graph 530b and copy Knowledge Cell 800bd into the inserted Knowledge Cell 800hd. The system may also create Connection 853h3 between Knowledge Cell 800hc and Knowledge Cell 800hd with occurrence count of 1 and weight calculated based on the occurrence count as previously described. The system may also update weights of other outgoing Connections 853 (one in this example) originating from Knowledge Cell 800hc as previously described. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800be from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Graph 530b. In the case that a substantially similar match is not found, the system may insert Knowledge Cell 800he into Graph 530b and copy Knowledge Cell 800be into the inserted Knowledge Cell 800he. The system may also create Connection 853h4 between Knowledge Cell 800hd and Knowledge Cell 800he with occurrence count of 1 and weight of 1. Applying any additional Knowledge Cells 800 from Knowledge Structuring Unit 520 onto Graph 530b follows similar logic or process as the above-described.

Referring to Fig. 23, an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Collection of Sequences 530c is illustrated. Collection of Sequences 530c comprises the functionality for storing one or more Sequences 533. Sequence 533 comprises the functionality for storing any number of Knowledge Cells 800. For example, Knowledge Structuring Unit 520 structures or generates Knowledge Cells 800 and the system applies them onto Collection of Sequences 530c, thereby implementing learning Device's 98 operation in circumstances including objects with various properties. The system can perform collective Similarity Comparisons 125 of Knowledge Cells 800 from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Sequences 533 of Collection of Sequences 530c to find a Sequence 533 comprising Knowledge Cells 800 that are collectively substantially similar to the Knowledge Cells 800 from Knowledge Structuring Unit 520. If Sequence 533 comprising such collectively substantially similar Knowledge Cells 800 is not found in Collection of Sequences 530c, the system

may create a new Sequence 533 comprising the Knowledge Cells 800 from Knowledge Structuring Unit 520 and insert (i.e. copy, store, etc.) the new Sequence 533 into Collection of Sequences 530c. On the other hand, if Sequence 533 comprising collectively substantially similar Knowledge Cells 800 is found in Collection of Sequences 530c, the system may optionally omit inserting the Knowledge Cells 800 from Knowledge Structuring Unit 520 into Collection of Sequences 530c as inserting a similar Sequence 533 may not add much or any additional knowledge. This approach can save storage resources and limit the number of Knowledge Cells 800 that may later need to be processed or compared. In some aspects, a Sequence 533 may include Knowledge Cells 800 relating to a single operation of Device 98. In other aspects, a Sequence 533 may include Knowledge Cells 800 relating to a part of an operation of Device 98. In further aspects, one or more long Sequences 533 each including Knowledge Cells 800 of multiple operations of Device 98 can be utilized. In one example, Knowledge Cells 800 of all operations can be stored in a single long Sequence 533 in which case Collection of Sequences 530c as a separate element can be omitted. In another example, Knowledge Cells 800 of multiple operations can be included in a plurality of long Sequences 533 such as hourly, daily, weekly, monthly, yearly, or other periodic or other Sequences 533. Similarity Comparisons 125 can be performed by traversing the one or more long Sequences 533 to find a match or substantially similar match. For instance, the system can perform collective Similarity Comparisons 125 of Knowledge Cells 800 from Knowledge Structuring Unit 520 with Knowledge Cells 800 in subsequences of a long Sequence 533 in incremental or other traversing pattern to find a subsequence comprising Knowledge Cells 800 that are collectively substantially similar to the Knowledge Cells 800 from Knowledge Structuring Unit 520. The incremental traversing pattern may start from one end of a long Sequence 533 and move the comparison subsequence up or down one or any number of incremental Knowledge Cells 800 at a time. Other traversing patterns or methods can be employed such as starting from the middle of the Sequence 533 and subdividing the resulting sub-sequences in a recursive pattern, or any other traversing pattern or method. If a subsequence comprising collectively substantially similar Knowledge Cells 800 is not found in the long Sequence 533, the system may concatenate or append the Knowledge Cells 800 from Knowledge Structuring Unit 520 to the long Sequence 533. In further aspects, Connections 853 can optionally be used in Sequence 533 to connect Knowledge Cells 800. For example, a Knowledge Cell 800 can be connected not only with a next Knowledge Cell 800 in the Sequence 533, but also with any other Knowledge Cell 800 in the Sequence 533, thereby creating alternate routes or shortcuts through the Sequence 533. Any number of Connections 853 connecting any Knowledge Cells 800 can be utilized. Any of the previously described and/or other techniques for comparing, inserting, updating, and/or other operations on Knowledge Cells 800, Connections 853, and/or other elements can similarly be utilized in Sequences 533 and/or Collection of Sequences 530c.

In some embodiments, various elements and/or techniques can be utilized in the aforementioned substantial similarity determinations with respect to collectively compared Knowledge Cells 800 and/or other elements. In some aspects, substantial similarity of collectively compared Knowledge Cells 800 can be determined based on similarities or similarity indexes of the individually compared Knowledge Cells 800. In one example, an average of similarities or similarity indexes of individually compared Knowledge Cells 800 can be used to determine similarity of collectively compared Knowledge Cells 800. In another example, a weighted average of similarities or similarity indexes of individually compared Knowledge Cells 800 can be used to determine similarity of collectively compared Knowledge Cells 800. For instance, to affect the weighting of collective similarity, a higher weight or

importance (i.e. importance index, etc.) can be assigned to the similarities or similarity indexes of some Knowledge Cells 800 and lower for other Knowledge Cells 800. Any higher or lower weight or importance assignment can be implemented. In other aspects, any of the previously described or other thresholds for substantial similarity of individually compared elements can similarly be utilized for collectively compared elements. In one example, substantial similarity of collectively compared Knowledge Cells 800 can be achieved when their collective similarity or similarity index exceeds a similarity threshold. In another example, substantial similarity of collectively compared Knowledge Cells 800 can be achieved when at least a threshold number or percentage of Knowledge Cells 800 from the collectively compared Knowledge Cells 800 match or substantially match. Similarly, substantial similarity of collectively compared Knowledge Cells 800 can be achieved when a number or percentage of matching or substantially matching Knowledge Cells 800 from the collectively compared Knowledge Cells 800 exceeds a threshold. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. Similar elements and/or techniques as the aforementioned can be used for similarity determinations of other collectively compared elements such as Collections of Object Representations 525, Object Representations 625, Object Properties 630, Instruction Sets 526, Extra Info 527, and/or others. Similarity determinations of collectively compared elements may include any features, functionalities, and embodiments of Similarity Comparison 125, and vice versa.

Any of the previously described data structures or arrangements of Knowledge Cells 800 such as Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, and/or others can be used alone, or in combination with each other or with other elements, in alternate embodiments. In one example, a path in Neural Network 530a or Graph 530b may include its own separate sequence of Knowledge Cells 800 that are not interconnected with Knowledge Cells 800 in other paths. In another example, a part of a path in Neural Network 530a or Graph 530b may include a sequence of Knowledge Cells 800 interconnected with Knowledge Cells 800 in other paths, whereas, another part of the path may include its own separate sequence of Knowledge Cells 800 that are not interconnected with Knowledge Cells 800 in other paths. Any other combinations or arrangements of Knowledge Cells 800 can be implemented.

Referring to Fig. 24, an embodiment of determining anticipatory Instruction Sets 526 from a single Knowledge Cell 800 is illustrated. Knowledge Cell 800 may be part of a Knowledgebase 530 (i.e. Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.) such as Collection of Knowledge Cells 530d. Decision-making Unit 540 comprises the functionality for anticipating or determining a device's operation in circumstances including objects with various properties. Decision-making Unit 540 comprises the functionality for anticipating or determining Instruction Sets 526 to be used or executed in Device's 98 autonomous operation. In some aspects, Instruction Sets 526 anticipated or determined to be used or executed in Device's 98 autonomous operation may be referred to as anticipatory Instruction Sets 526, alternate Instruction Sets 526, and/or other suitable name or reference. Therefore, these terms can be used interchangeably herein depending on context. Decision-making Unit 540 also comprises other disclosed functionalities.

In some aspects, Decision-making Unit 540 may anticipate or determine Instruction Sets 526 (i.e. anticipatory Instruction Sets 526, etc.) for autonomous Device 98 operation by performing Similarity Comparisons 125 of incoming Collections of Object Representations 525 or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Knowledgebase 530 (i.e.

Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.). A Knowledge Cell 800 includes knowledge (i.e. one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527, etc.) of how Device 98 operated in a circumstance including objects with various properties as previously described. When one or more Collections of Object Representations 525 representing objects with similar properties are received in the future, Decision-making Unit 540 can anticipate the Instruction Sets 526 (i.e. anticipatory Instruction Sets 526, etc.) previously learned in a similar circumstance, thereby enabling autonomous Device 98 operation. In some aspects, Decision-making Unit 540 can perform Similarity Comparisons 125 of incoming Collections of Object Representations 525 from Object Processing Unit 93 with Collections of Object Representations 525 from Knowledge Cells 800 in Knowledgebase 530 (i.e. Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.). If one or more substantially similar Collections of Object Representations 525 or portions thereof are found in a Knowledge Cell 800 from Knowledgebase 530, Instruction Sets 526 (i.e. anticipatory Instruction Sets 526, etc.) for autonomous Device 98 operation can be anticipated in Instruction Sets 526 correlated with the one or more Collections of Object Representations 525 from the Knowledge Cell 800. In some designs, subsequent one or more Instruction Sets 526 for autonomous Device 98 operation can be anticipated in Instruction Sets 526 correlated with subsequent Collections of Object Representations 525 from the Knowledge Cell 800 or other Knowledge Cells 800, thereby anticipating not only current, but also additional future Instruction Sets 526. Although, Extra Info 527 is not shown in this and/or other figures for clarity of illustration, it should be noted that any Collection of Object Representations 525, Instruction Set 526, and/or other element may include or be associated with Extra Info 527 and that Decision-making Unit 540 can utilize Extra Info 527 for enhanced decision making.

For example, Decision-making Unit 540 can perform Similarity Comparison 125 of Collection of Object Representations 525l1 or portions thereof from Object Processing Unit 93 with Collection of Object Representations 525a1 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a1 or portions thereof from Knowledge Cell 800oa may be found substantially similar. Decision-making Unit 540 can anticipate Instruction Sets 526a1-526a3 correlated with Collection of Object Representations 525a1, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparison 125 of Collection of Object Representations 525l2 or portions thereof from Object Processing Unit 93 with Collection of Object Representations 525a2 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a2 or portions thereof from Knowledge Cell 800oa may be found substantially similar. Decision-making Unit 540 can anticipate Instruction Set 526a4 correlated with Collection of Object Representations 525a2, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparison 125 of Collection of Object Representations 525l3 or portions thereof from Object Processing Unit 93 with Collection of Object Representations 525a3 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a3 or portions thereof from Knowledge Cell 800oa may be found substantially similar. Decision-making Unit 540 may not anticipate any Instruction Sets 526 since none are correlated with Collection of Object Representations 525a3. Decision-making Unit 540 can then perform Similarity Comparison 125 of Collection of Object Representations 525l4 or portions thereof from Object Processing Unit 93 with Collection of Object Representations 525a4 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a4 or portions thereof from Knowledge Cell 800oa may not be found substantially similar. Decision-making Unit 540 can then

perform Similarity Comparison 125 of Collection of Object Representations 525l5 or portions thereof from Object Processing Unit 93 with Collection of Object Representations 525a5 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a5 or portions thereof from Knowledge Cell 800oa may not be found substantially similar. Decision-making Unit 540 can implement similar logic or process for any additional Collections of Object Representations 525 from Object Processing Unit 93, and so on.

It should be understood that any of the described elements and/or techniques in the foregoing example can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells 800 or elements (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof would be affected accordingly. In one example, Extra Info 527 can be included in the Similarity Comparisons 125 as previously described. In another example, as history of incoming Collections of Object Representations 525 becomes available, Decision-making Unit 540 can perform collective Similarity Comparisons 125 of the history of Collections of Object Representations 525 or portions thereof from Object Processing Unit 93 with subsequences of Collections of Object Representations 525 or portions thereof from Knowledge Cell 800. In a further example, the described comparisons in a single Knowledge Cell 800 may be performed on any number of Knowledge Cells 800 sequentially or in parallel. Parallel processors such as a plurality of Processors 11 or cores thereof can be utilized for such parallel processing. In a further example, various arrangements of Collections of Object Representations 525 and/or other elements in a Knowledge Cell 800 can be utilized as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to Fig. 25, an embodiment of determining anticipatory Instruction Sets 526 by traversing a single Knowledge Cell 800 is illustrated. Knowledge Cell 800 may be part of a Knowledgebase 530 (i.e. Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.) such as Collection of Knowledge Cells 530d. For example, Decision-making Unit 540 can perform Similarity Comparison 125 of Collection of Object Representations 525l1 or portions thereof from Object Processing Unit 93 with Collection of Object Representations 525a1 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a1 or portions thereof from Knowledge Cell 800oa may not be found substantially similar. Decision-making Unit 540 can then perform Similarity Comparison 125 of Collection of Object Representations 525l1 or portions thereof from Object Processing Unit 93 with Collection of Object Representations 525a2 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a2 or portions thereof from Knowledge Cell 800oa may not be found substantially similar. Decision-making Unit 540 can then perform Similarity Comparison 125 of Collection of Object Representations 525l1 or portions thereof from Object Processing Unit 93 with Collection of Object Representations 525a3 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a3 or portions thereof from Knowledge Cell 800oa may be found substantially similar. Decision-making Unit 540 may not anticipate any Instruction Sets 526 since none are correlated with Collection of Object Representations 525a3. Decision-making Unit 540 can then perform Similarity Comparison 125 of Collection of Object Representations 525l2 or portions thereof from Object Processing Unit 93 with Collection of Object Representations 525a4 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a4 or portions thereof from Knowledge Cell 800oa may be found substantially similar. Decision-making Unit 540 can anticipate Instruction Sets 526a5-

526a6 correlated with Collection of Object Representations 525a4, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparison 125 of Collection of Object Representations 525l3 or portions thereof from Object Processing Unit 93 with Collection of Object Representations 525a5 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a5 or portions thereof from Knowledge Cell 800oa may be found substantially similar. Decision-making Unit 540 may not anticipate any Instruction Sets 526 since none are correlated with Collection of Object Representations 525a5. Decision-making Unit 540 can implement similar logic or process for any additional Collections of Object Representations 525 from Object Processing Unit 93, and so on.

It should be understood that any of the described elements and/or techniques in the foregoing example can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells 800 or elements (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof would be affected accordingly. In one example, Extra Info 527 can be included in the Similarity Comparisons 125 as previously described. In another example, as history of incoming Collections of Object Representations 525 becomes available, Decision-making Unit 540 can perform collective Similarity Comparisons 125 of the history of Collections of Object Representations 525 or portions thereof from Object Processing Unit 93 with subsequences of Collections of Object Representations 525 or portions thereof from Knowledge Cell 800. In a further example, traversing may be performed in incremental traversing pattern such as starting from one end of Knowledge Cell 800 and moving the comparison subsequence up or down the list one or any number of incremental Collections of Object Representations 525 at a time. Other traversing patterns or methods can be employed such as starting from the middle of the Knowledge Cell 800 and subdividing the resulting subsequence in a recursive pattern, or any other traversing pattern or method. In a further example, the described traversing of a single Knowledge Cell 800 may be performed on any number of Knowledge Cells 800 sequentially or in parallel. Parallel processors such as a plurality of Processors 11 or cores thereof can be utilized for such parallel processing. In a further example, various arrangements of Collections of Object Representations 525 and/or other elements in a Knowledge Cell 800 can be utilized as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to Fig. 26, an embodiment of determining anticipatory Instruction Sets 526 using collective similarity comparisons is illustrated. For example, Decision-making Unit 540 can perform Similarity Comparisons 125 of Collection of Object Representations 525l1 or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Collection of Knowledge Cells 530d. Collection of Object Representations 525c1 or portions thereof from Knowledge Cell 800rc may be found substantially similar with highest similarity. Decision-making Unit 540 can anticipate any Instruction Sets 526 (not shown) correlated with Collection of Object Representations 525c1, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform collective Similarity Comparisons 125 of Collections of Object Representations 525l1-525l2 or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Collection of Knowledge Cells 530d. Collections of Object Representations 525c1-525c2 or portions thereof from Knowledge Cell 800rc may be found

substantially similar with highest similarity. Decision-making Unit 540 can anticipate any Instruction Sets 526 (not shown) correlated with Collection of Object Representations 525c2, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform collective Similarity Comparisons 125 of Collections of Object Representations 525l1-525l3 or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Collection of Knowledge Cells 530d. Collections of Object Representations 525d1-525d3 or portions thereof from Knowledge Cell 800rd may be found substantially similar with highest similarity. Decision-making Unit 540 can anticipate any Instruction Sets 526 (not shown) correlated with Collection of Object Representations 525d3, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform collective Similarity Comparisons 125 of Collections of Object Representations 525l1-525l4 or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Collection of Knowledge Cells 530d. Collections of Object Representations 525d1-525d4 or portions thereof from Knowledge Cell 800rd may be found substantially similar with highest similarity. Decision-making Unit 540 can anticipate any Instruction Sets 526 (not shown) correlated with Collection of Object Representations 525d4, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform collective Similarity Comparisons 125 of Collections of Object Representations 525l1-525l5 or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Collection of Knowledge Cells 530d. Collections of Object Representations 525d1-525d5 or portions thereof from Knowledge Cell 800rd may be found substantially similar with highest similarity. Decision-making Unit 540 can anticipate any Instruction Sets 526 (not shown) correlated with Collection of Object Representations 525d5, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can implement similar logic or process for any additional Collections of Object Representations 525 from Object Processing Unit 93, and so on.

In some embodiments, various elements and/or techniques can be utilized in the aforementioned similarity determinations with respect to collectively compared Collections of Object Representations 525 and/or other elements. In some aspects, similarity of collectively compared Collections of Object Representations 525 can be determined based on similarities or similarity indexes of the individually compared Collections of Object Representations 525. In one example, an average of similarities or similarity indexes of individually compared Collections of Object Representations 525 can be used to determine similarity of collectively compared Collections of Object Representations 525. In another example, a weighted average of similarities or similarity indexes of individually compared Collections of Object Representations 525 can be used to determine similarity of collectively compared Collections of Object Representations 525. For instance, to affect the weighting of collective similarity, a higher weight or importance (i.e. importance index, etc.) can be assigned to the similarities or similarity indexes of some (i.e. more substantive or larger, etc.) Collections of Object Representations 525 and lower for other (i.e. less substantive or smaller, etc.) Collections of Object Representations 525. Any other higher or lower weight or importance assignment can be implemented. In other aspects, any of the previously described or other thresholds for substantial similarity of individually compared elements can be similarly utilized for collectively compared elements. In one example, substantial similarity of collectively compared Collections of Object Representations 525 can be achieved when their collective similarity or similarity index exceeds a similarity threshold. In another example, substantial similarity of collectively compared Collections of Object Representations 525 can be achieved when at

least a threshold number or percentage of Collections of Object Representations 525 or portions thereof from the collectively compared Collections of Object Representations 525 match or substantially match. Similarly, substantial similarity of collectively compared Collections of Object Representations 525 can be achieved when a number or percentage of matching or substantially matching Collections of Object Representations 525 or portions thereof from the collectively compared Collections of Object Representations 525 exceeds a threshold. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. Similar elements and/or techniques as the aforementioned can be used for similarity determinations of other collectively compared elements such as Object Representations 625, Object Properties 630, Instruction Sets 526, Extra Info 527, Knowledge Cells 800, and/or others. Similarity determinations of collectively compared elements may include any features, functionalities, and embodiments of Similarity Comparison 125, and vice versa.

It should be understood that any of the described elements and/or techniques in the foregoing example can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells 800 or elements (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info 527 in Similarity Comparisons 125, traversing of Knowledge Cells 800 or other elements, using history of Collections of Object Representations 525 or Knowledge Cells 800 for collective Similarity Comparisons 125, using various arrangements of Collections of Object Representations 525 and/or other elements in a Knowledge Cell 800, and/or others can similarly be utilized in this example. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to Fig. 27, an embodiment of determining anticipatory Instruction Sets 526 using Neural Network 530a is illustrated. In some aspects, determining anticipatory Instruction Sets 526 using Neural Network 530a may include selecting a path of Knowledge Cells 800 or elements (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof through Neural Network 530a. Decision-making Unit 540 can utilize various elements and/or techniques for selecting a path through Neural Network 530a. Although, these elements and/or techniques are described with respect to Neural Network 530a below, they can similarly be used in any Knowledgebase 530 (i.e. Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.) as applicable.

In some embodiments, Decision-making Unit 540 can utilize similarity index in selecting Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof in a path through Neural Network 530a. For instance, similarity index may indicate how well one Knowledge Cell 800 or portions thereof are matched with another Knowledge Cell 800 or portions thereof as previously described. In one example, Decision-making Unit 540 may select a Knowledge Cell 800 comprising one or more Collections of Object Representations 525 with highest similarity index even if Connection 853 pointing to that Knowledge Cell 800 has less than the highest weight. Therefore, similarity index or other such element or parameter can override or disregard the weight of a Connection 853 or other element. In another example, Decision-making Unit 540 may select a Knowledge Cell 800 comprising one or more Collections of Object Representations 525 whose similarity

index is higher than or equal to a weight of Connection 853 pointing to that Knowledge Cell 800. In a further example, Decision-making Unit 540 may select a Knowledge Cell 800 comprising one or more Collections of Object Representations 525 whose similarity index is lower than or equal to a weight of Connection 853 pointing to that Knowledge Cell 800. Similarity index can be set to be more, less, or equally important than a weight of a Connection 853.

In some embodiments, Decision-making Unit 540 can utilize Connections 853 in selecting Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof in a path through Neural Network 530a. In some aspects, Decision-making Unit 540 can take into account weights of Connections 853 among the interconnected Knowledge Cells 800 in choosing from which Knowledge Cell 800 to compare one or more Collections of Object Representations 525 first, second, third, and so on. Specifically, for instance, Decision-making Unit 540 can perform Similarity Comparisons 125 with one or more Collections of Object Representations 525 from Knowledge Cell 800 pointed to by the highest weight Connection 853 first, Collections of Object Representations 525 from Knowledge Cell 800 pointed to by the second highest weight Connection 853 second, and so on. In other aspects, Decision-making Unit 540 can stop performing Similarity Comparisons 125 as soon as it finds one or more substantially similar Collections of Object Representations 525 in an interconnected Knowledge Cell 800. In further aspects, Decision-making Unit 540 may only follow the highest weight Connection 853 to arrive at a Knowledge Cell 800 comprising one or more Collections of Object Representations 525 to be compared, thereby disregarding Connections 853 with less than the highest weight. In further aspects, Decision-making Unit 540 may ignore weights and/or other parameters of Connections 853. In further aspects, Decision-making Unit 540 may ignore Connections 853.

In some embodiments, Decision-making Unit 540 can utilize a bias to adjust similarity index, weight of a Connection 853, and/or other element or parameter used in selecting Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof in a path through Neural Network 530a. In one example, Decision-making Unit 540 may select a Knowledge Cell 800 comprising one or more Collections of Object Representations 525 whose similarity index multiplied by or adjusted for a bias is higher than or equal to a weight of Connection 853 pointing to that Knowledge Cell 800. In another example, Decision-making Unit 540 may select a Knowledge Cell 800 comprising one or more Collections of Object Representations 525 whose similarity index multiplied by or adjusted for a bias is lower than or equal to a weight of Connection 853 pointing to that Knowledge Cell 800. In a further example, bias can be used to resolve deadlock situations where similarity index is equal to a weight of a Connection 853. In some aspects, bias can be expressed in percentages such as 0.3 percent, 1.2 percent, 25.7 percent, 79.8 percent, 99.9 percent, 100.1 percent, 155.4 percent, 298.6 percent, 1105.5 percent, and so on. For example, a bias below 100 percent decreases an element or parameter to which it is applied, a bias equal to 100 percent does not change the element or parameter to which it is applied, and a bias higher than 100 percent increases the element or parameter to which it is applied. In general, any amount of bias can be utilized depending on implementation. Bias can be applied to one or more of a weight of a Connection 853, similarity index, any other element or parameter, and/or all or any combination of them. Also, different biases can be applied to each of a weight of a Connection 853, similarity index, or any other element or parameter. For example, 30 percent bias can be applied to similarity index and 15 percent bias can be applied to a weight of a Connection 853. Also, different biases can be applied to various Layers 854 of Neural Network 530a, and/or other

disclosed elements. Bias can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input.

Any other element and/or technique can be utilized in selecting Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof in a path through Neural Network 530a.

In some embodiments, Neural Network 530a may include knowledge (i.e. interconnected Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527, etc.) of how Device 98 operated in circumstances including objects with various properties. In some aspects, determining anticipatory Instruction Sets 526 using Neural Network 530a may include selecting a path of Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof through Neural Network 530a. Individual and/or collective Similarity Comparisons 125 can be used to determine substantial similarity of the individually and/or collectively compared Collections of Object Representations 525 or portions thereof. Substantial similarity may be used primarily for selecting a path through Neural Network 530a, whereas, weight of any Connection 853 may be used secondarily or not at all.

For example, Decision-making Unit 540 can perform Similarity Comparisons 125 of Collections of Object Representations 525a1-525an or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from one or more Knowledge Cells 800 in Layer 854a (or any other one or more Layers 854, etc.). Collections of Object Representations 525 or portions thereof from Knowledge Cell 800ta may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object Representations 525b1-525bn or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from one or more Knowledge Cells 800 in Layer 854b interconnected with Knowledge Cell 800ta. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800tb may be found collectively substantially similar with highest similarity, thus, Decision-making Unit 540 may follow Connection 853t1 disregarding its less than highest weight. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Since Connection 853t2 is the only connection from Knowledge Cell 800tb, Decision-making Unit 540 may follow Connection 853t2 and perform Similarity Comparisons 125 of Collections of Object Representations 525c1-525cn or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cell 800tc in Layer 854c. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800tc may be found collectively substantially similar. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object

Representations 525d1-525dn or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from one or more Knowledge Cells 800 in Layer 854d interconnected with Knowledge Cell 800tc. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800td may be found collectively substantially similar with highest similarity, thus, Decision-making Unit 540 may follow Connection 853t3. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object Representations 525e1-525en or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from one or more Knowledge Cells 800 in Layer 854e interconnected with Knowledge Cell 800td. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800te may be found collectively substantially similar with highest similarity, thus, Decision-making Unit 540 may follow Connection 853t4. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can implement similar logic or process for any additional Collections of Object Representations 525 from Object Processing Unit 93, and so on.

The foregoing exemplary embodiment provides an example of utilizing a combination of collective Similarity Comparisons 125, individual Similarity Comparisons 125, Connections 853, and/or other elements or techniques. It should be understood that any of these elements and/or techniques can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof in a path through Neural Network 530a would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info 527 in Similarity Comparisons 125, traversing of Knowledge Cells 800 or other elements, using history of Collections of Object Representations 525 or Knowledge Cells 800 for collective Similarity Comparisons 125, using various arrangements of Collections of Object Representations 525 and/or other elements in a Knowledge Cell 800, and/or others can similarly be utilized in this example. These elements and/or techniques can similarly be utilized in Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, and/or other data structures or arrangements. In some aspects, instead of anticipating Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar streams of Collections of Object Representations 525. In other aspects, any time that substantial similarity or other similarity threshold is not achieved in compared Collections of Object Representations 525 or portions thereof from any of the Knowledge Cells 800, Decision-making Unit 540 can decide to look for a substantially or otherwise similar Collections of Object Representations 525 or portions thereof in Knowledge Cells 800 elsewhere in Neural Network 530a such as in any Layer 854 subsequent to a current Layer 854, in the first Layer 854, in the entire Neural Network 530a, and/or others, even if such Knowledge Cell 800 may be unconnected with a prior Knowledge Cell 800. It should be noted that any of Collections of Object Representations 525a1-525an, Collections of Object Representations 525b1-525bn, Collections of Object Representations 525c1-525cn, Collections of Object

Representations 525d1-525dn, Collections of Object Representations 525e1-525en, etc. may include one Collection of Object Representations 525 or a stream of Collections of Object Representations 525. It should also be noted that any Knowledge Cell 800 may include one Collection of Object Representations 525 or a stream of Collections of Object Representations 525 as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to Fig. 28, an embodiment of determining anticipatory Instruction Sets 526 using Graph 530b is illustrated. Graph 530b may include knowledge (i.e. interconnected Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527, etc.) of how Device 98 operated in circumstances including objects with various properties. In some aspects, determining anticipatory Instruction Sets 526 using Graph 530b may include selecting a path of Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof through Graph 530b. Individual and/or collective Similarity Comparisons 125 can be used to determine substantial similarity of the individually and/or collectively compared Collections of Object Representations 525 or portions thereof. Substantial similarity may be used primarily for selecting a path through Graph 530b, whereas, weight of any Connection 853 may be used secondarily or not at all.

For example, Decision-making Unit 540 can perform Similarity Comparisons 125 of Collections of Object Representations 525a1-525an or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from one or more Knowledge Cells 800 in Graph 530b. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800ua may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object Representations 525b1-525bn or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from one or more Knowledge Cells 800 in Graph 530b interconnected with Knowledge Cell 800ua by outgoing Connections 853. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800ub may be found collectively substantially similar with highest similarity, thus, Decision-making Unit 540 may follow Connection 853u1 disregarding its less than highest weight. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object Representations 525c1-525cn or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from one or more Knowledge Cells 800 in Graph 530b interconnected with Knowledge Cell 800ub by outgoing Connections 853. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800uc may be found collectively substantially similar with highest similarity, thus, Decision-making Unit 540 may follow Connection 853u2 disregarding its less than highest weight. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-

making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Since Connection 853u3 is the only connection from Knowledge Cell 800uc, Decision-making Unit 540 may follow Connection 853u3 and perform Similarity Comparisons 125 of Collections of Object Representations 525d1-525dn or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cell 800ud in Graph 530b. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800ud may be found collectively substantially similar. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object Representations 525e1-525en or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from one or more Knowledge Cells 800 in Graph 530b interconnected with Knowledge Cell 800ud by outgoing Connections 853. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800ue may be found collectively substantially similar with highest similarity, thus, Decision-making Unit 540 may follow Connection 853u4. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can implement similar logic or process for any additional Collections of Object Representations 525 from Object Processing Unit 93, and so on.

The foregoing exemplary embodiment provides an example of utilizing a combination of collective Similarity Comparisons 125, individual Similarity Comparisons 125, Connections 853, and/or other elements or techniques. It should be understood that any of these elements and/or techniques can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof in a path through Graph 530b would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info 527 in Similarity Comparisons 125, traversing of Knowledge Cells 800 or other elements, using history of Collections of Object Representations 525 or Knowledge Cells 800 in collective Similarity Comparisons 125, using various arrangements of Collections of Object Representations 525 and/or other elements in a Knowledge Cell 800, and/or others can similarly be utilized in this example. These elements and/or techniques can similarly be utilized in Neural Network 530a, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, and/or other data structures or arrangements. In some aspects, instead of anticipating Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially matching streams of Collections of Object Representations 525. In other aspects, any time that substantial similarity or other similarity threshold is not achieved in compared Collections of Object Representations 525 or portions thereof of any of the Knowledge Cells 800, Decision-making Unit 540 can decide to look for a substantially or otherwise similar Collections of Object Representations 525 or portions thereof in Knowledge Cells 800 elsewhere in Graph 530b even if such Knowledge Cell 800 may be unconnected with a prior Knowledge Cell 800. It should be

noted that any of Collections of Object Representations 525a1-525an, Collections of Object Representations 525b1-525bn, Collections of Object Representations 525c1-525cn, Collections of Object Representations 525d1-525dn, Collections of Object Representations 525e1-525en, etc. may include one Collection of Object Representations 525 or a stream of Collections of Object Representations 525. It should also be noted that any Knowledge Cell 800 may include one Collection of Object Representations 525 or a stream of Collections of Object Representations 525 as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to Fig. 29, an embodiment of determining anticipatory Instruction Sets 526 using Collection of Sequences 530c is illustrated. Collection of Sequences 530c may include knowledge (i.e. sequences of Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527, etc.) of how Device 98 operated in circumstances including objects with various properties. In some aspects, determining anticipatory Instruction Sets 526 for autonomous Device 98 operation using Collection of Sequences 530c may include selecting a Sequence 533 of Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof from Collection of Sequences 530c. Individual and/or collective Similarity Comparisons 125 can be used to determine substantial similarity of the individually and/or collectively compared Collections of Object Representations 525 or portions thereof.

For example, Decision-making Unit 540 can perform Similarity Comparisons 125 of Collections of Object Representations 525a1-525an or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in one or more Sequences 533 of Collection of Sequences 530c. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800ca in Sequence 533wc may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object Representations 525a1-525an and 525b1-525bn or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Sequences 533 of Collection of Sequences 530c. Collections of Object Representations 525 or portions thereof from Knowledge Cells 800ca-800cb in Sequence 533wc may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object Representations 525a1-525an, 525b1-525bn, and 525c1-525cn or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Sequences 533 of Collection of Sequences 530c. Collections of Object Representations 525 or portions thereof from Knowledge Cells 800da-800dc in Sequence 533wd may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-

making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object Representations 525a1-525an, 525b1-525bn, 525c1-525cn, and 525d1-525dn or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Sequences 533 of Collection of Sequences 530c. Collections of Object Representations 525 or portions thereof from Knowledge Cells 800da-800dd in Sequence 533wd may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object Representations 525a1-525an, 525b1-525bn, 525c1-525cn, 525d1-525dn, and 525e1-525en or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Sequences 533 of Collection of Sequences 530c. Collections of Object Representations 525 or portions thereof from Knowledge Cells 800da-800de in Sequence 533wd may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can implement similar logic or process for any additional Collections of Object Representations 525 from Object Processing Unit 93, and so on.

The foregoing exemplary embodiment provides an example of utilizing a combination of collective Similarity Comparisons 125, individual Similarity Comparisons 125, and/or other elements or techniques. It should be understood that any of these elements and/or techniques can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Sequence 533 of Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info 527 in Similarity Comparisons 125, traversing of Knowledge Cells 800 or other elements, using history of Collections of Object Representations 525 or Knowledge Cells 800 in collective Similarity Comparisons 125, using various arrangements of Collections of Object Representations 525 and/or other elements in a Knowledge Cell 800, and/or others can similarly be utilized in this example. These elements and/or techniques can similarly be utilized in Neural Network 530a, Graph 530b, Collection of Knowledge Cells 530d, and/or other data structures or arrangements. In some aspects, instead of anticipating Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially matching streams of Collections of Object Representations 525. In other aspects, any time that substantial similarity or other similarity threshold is not achieved in compared Collections of Object Representations 525 or portions thereof from any of the Knowledge Cells 800, Decision-making Unit 540 can decide to look for a substantially or otherwise similar Collections of Object Representations 525 or portions thereof in Knowledge Cells 800 elsewhere in Collection of Sequences 530c such as in different Sequences 533. It should be noted that any of Collections of Object Representations 525a1-525an, Collections of

Object Representations 525b1-525bn, Collections of Object Representations 525c1-525cn, Collections of Object Representations 525d1-525dn, Collections of Object Representations 525e1-525en, etc. may include one Collection of Object Representations 525 or a stream of Collections of Object Representations 525. It should also be noted that any Knowledge Cell 800 may include one Collection of Object Representations 525 or a stream of Collections of Object Representations 525 as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring now to Modification Interface 130. Modification Interface 130 comprises the functionality for modifying execution and/or functionality of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. Modification Interface 130 comprises the functionality for modifying execution and/or functionality of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element at runtime. Modification Interface 130 comprises the functionality for modifying execution and/or functionality of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element based on anticipatory Instruction Sets 526. In one example, Modification Interface 130 comprises the functionality to access, modify, and/or perform other manipulations on runtime engine/environment, virtual machine, operating system, compiler, just-in-time (JIT) compiler, interpreter, translator, execution stack, file, object, data structure, and/or other computing system elements. In another example, Modification Interface 130 comprises the functionality to access, modify, and/or perform other manipulations on memory, storage, bus, interfaces, and/or other computing system elements. In a further example, Modification Interface 130 comprises the functionality to access, modify, and/or perform other manipulations on Processor 11 registers and/or other Processor 11 elements. In a further example, Modification Interface 130 comprises the functionality to access, modify, and/or perform other manipulations on inputs and/or outputs of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. In a further example, Modification Interface 130 comprises the functionality to access, create, delete, modify, and/or perform other manipulations on functions, methods, procedures, routines, subroutines, and/or other elements of Application Program 18. In a further example, Modification Interface 130 comprises the functionality to access, create, delete, modify, and/or perform other manipulations on source code, bytecode, compiled, interpreted, or otherwise translated code, machine code, and/or other code. In a further example, Modification Interface 130 comprises the functionality to access, create, delete, modify, and/or perform other manipulations on values, variables, parameters, and/or other data or information. Modification Interface 130 comprises any features, functionalities, and embodiments of Acquisition Interface 120, and vice versa. Modification Interface 130 also comprises other disclosed functionalities.

Modification Interface 130 can employ various techniques for modifying execution and/or functionality of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. In some aspects, some of the previously described techniques and/or tools can be utilized. Code instrumentation, for instance, may involve inserting additional code, overwriting or rewriting existing code, and/or branching to a separate segment of code in Application Program 18 as previously described. For example, instrumented code may include the following:

```
Device1.moveLeft(23);
modifyApplication();
```

In the above sample code, instrumented call to Modification Interface's 130 function (i.e. modifyApplication(), etc.) can be placed after a function (i.e. Device1.moveLeft(23), etc.) of Application Program 18. Similar call to an

application modifying function can be placed after or before some or all functions/routines/subroutines, some or all lines of code, some or all statements, some or all instructions or instruction sets, some or all basic blocks, and/or some or all other code segments of Application Program 18. One or more application modifying function calls can be placed anywhere in Application Program's 18 code and can be executed at any points in Application Program's 18 execution. The application modifying function (i.e. modifyApplication(), etc.) may include Artificial Intelligence Unit 110-determined anticipatory Instruction Sets 526 that can modify execution and/or functionality of Application Program 18. In some embodiments, the previously described obtaining Application Program's 18 instruction sets, data, and/or other information as well as modifying execution and/or functionality of Application Program 18 can be implemented in a single function that performs both tasks (i.e. traceAndModifyApplication(), etc.).

In some embodiments, various computing systems and/or platforms may provide native tools for modifying execution and/or functionality of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. Independent vendors may provide tools with similar functionalities that can be utilized across different platforms. These tools enable a wide range of techniques or capabilities such as instrumentation, self-modifying code capabilities, dynamic code capabilities, branching, code rewriting, code overwriting, hot swapping, accessing and/or modifying objects or data structures, accessing and/or modifying functions/routines/subroutines, accessing and/or modifying variable or parameter values, accessing and/or modifying processor registers, accessing and/or modifying inputs and/or outputs, providing runtime memory access, and/or other capabilities. One of ordinary skill in art will understand that, while all possible variations of the techniques for modifying execution and/or functionality of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element are too voluminous to describe, these techniques are within the scope of this disclosure.

In one example, modifying execution and/or functionality of Application Program 18 can be implemented through utilizing metaprogramming techniques, which include applications that can self-modify or that can create, modify, and/or manipulate other applications. Self-modifying code, dynamic code, reflection, and/or other techniques can be used to facilitate metaprogramming. In some aspects, metaprogramming is facilitated through a programming language's ability to access and manipulate the internals of the runtime engine directly or via an API. In other aspects, metaprogramming is facilitated through dynamic execution of expressions (i.e. anticipatory Instruction Sets 526, etc.) that can be created and/or executed at runtime. In yet other aspects, metaprogramming is facilitated through application modification tools, which can perform modifications on an application regardless of whether the application's programming language enables metaprogramming capabilities. Some operating systems may protect an application loaded into memory by restricting access to the loaded application. This protection mechanism can be circumvented by utilizing operating system's, processor's, and/or other low level features or commands to unprotect the loaded application. For example, a self-modifying application may modify the in-memory image of itself. To do so, the application can obtain the in-memory address of its code. The application may then change the operating system's or platform's protection on this memory range allowing it to modify the code (i.e. insert anticipatory Instruction Sets 526, etc.). In addition to a self-modifying application, one application can utilize similar technique to modify another application. Linux mprotect command or similar commends of other operating systems can be used to change protection (i.e. unprotect, etc.) for a region of memory, for example. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program 18 can be

implemented through native capabilities of dynamic, interpreted, and/or scripting programming languages and/or platforms. Most of these languages and/or platforms can perform functionalities at runtime that static programming languages may perform during compilation. Dynamic, interpreted, and/or scripting languages provide native functionalities such as self-modification of code, dynamic code, extending the application, adding new code, extending objects and definitions, and/or other functionalities that can modify an application's execution and/or functionality at runtime. Examples of dynamic, interpreted, and/or scripting languages include Lisp, Perl, PHP, JavaScript, Ruby, Python, Smalltalk, Tcl, VBScript, and/or others. Similar functionalities can also be provided in languages such as Java, C, and/or others using reflection. Reflection includes the ability of an application to examine and modify the structure and behavior of the application at runtime. For example, JavaScript can modify its own code as it runs by utilizing Function object constructor as follows:

```
myFunc=new Function(arg1, arg2, argN, functionBody);
```

The sample code above causes a new function object to be created with the specified arguments and body. The body and/or arguments of the new function object may include new instruction sets (i.e. anticipatory Instruction Sets 526, etc.). The new function can be invoked as any other function in the original code. In another example, JavaScript can utilize eval method that accepts a string of JavaScript statements (i.e. anticipatory Instruction Sets 526, etc.) and execute them as if they were within the original code. An example of how eval method can be used to modify an application includes the following JavaScript code:

```
anticipatoryInstr = 'Device1.moveForward(27);';
if (anticipatoryInstr != "" && anticipatoryInstr != null)
{
    eval(anticipatoryInstr);
}
```

In the sample code above, Artificial Intelligence Unit 110 may generate anticipatory Instruction Set 526 (i.e. 'Device1.moveForward(27)' for moving a Device1 forward 27 units, etc.) and save it in anticipatoryInstr variable, which eval method can then execute. Lisp is another example of dynamic, interpreted, and/or scripting language that includes similar capabilities as previously described JavaScript. For example, Lisp's compile command can create a function at runtime, eval command may parse and evaluate an expression at runtime, and exec command may execute a given instruction set (i.e. string, etc.) at runtime. In another example, dynamic as well as some non-dynamic languages may provide macros, which combine code introspection and/or eval capabilities. In some aspects, macros can access inner workings of the compiler, interpreter, virtual machine, runtime environment/engine, and/or other components of the computing platform enabling the definition of language-like constructs and/or generation of a complete program or sections thereof. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program 18 can be implemented through dynamic code, dynamic class loading, reflection, and/or other native functionalities of a programming language or platform. In static applications or static programming, a class can be defined and/or loaded at compile time. Conversely, in dynamic applications or dynamic programming, a class can be loaded into a running environment at runtime. For example, Java Runtime Environment (JRE) may not require that all classes be loaded at compile time and class loading can occur when a class is first referenced at runtime. Dynamic class

loading enables inclusion or injection of on-demand code and/or functionalities at runtime. System provided or custom class loaders may enable loading of classes into the running environment. Custom class loaders can be created to enable custom functionalities such as, for example, specifying a remote location from which a class can be loaded. In addition to dynamic loading of a pre-defined class, a class can also be created at runtime. In some aspects, a class source code can be created at runtime. A compiler such as javac, com.sun.tools.javac.Main, javax.tools, javax.tools.JavaCompiler, and/or other packages can then be utilized to compile the source code. Javac, com.sun.tools.javac.Main, javax.tools, javax.tools.JavaCompiler, and/or other packages may include an interface to invoke Java compiler from within a running application. A Java compiler may accept source code in a file, string, object (i.e. Java String, StringBuffer, CharSequence, etc.) and/or other source, and may generate Java bytecode (i.e. class file, etc.). Once compiled, a class loader can then load the compiled class into the running environment. In other aspects, a tool such as Javaassist (i.e. Java programming assistant) can be utilized to enable an application to create or modify a class at runtime. Javassist may include a Java library that provides functionalities to create and/or manipulate Java bytecode of an application as well as reflection capabilities. Javassist may provide source-level and bytecode-level APIs. Using the source-level API, a class can be created and/or modified using only source code, which Javassist may compile seamlessly on the fly. Javassist source-level API can therefore be used without knowledge of Java bytecode specification. Bytecode-level API enables creating and/or editing a class bytecode directly. In yet other aspects, similar functionalities to the aforementioned ones may be provided in tools such as Apache Commons BCEL (Byte Code Engineering Library), ObjectWeb ASM, CGLIB (Byte Code Generation Library), and/or others. Once a dynamic code or class is created and loaded, reflection in high-level programming languages such as Java and/or others can be used to manipulate or change the runtime behavior of an application. Examples of reflective programming languages and/or platforms include Java, JavaScript, Smalltalk, Lisp, Python, .NET Common Language Runtime (CLR), Tcl, Ruby, Perl, PHP, Scheme, PL/SQL, and/or others. Reflection can be used in an application to access, examine, modify, and/or manipulate a loaded class and/or its elements. Reflection in Java can be implemented by utilizing a reflection API such as java.lang.Reflect package. The reflection API provides functionalities such as, for example, loading or reloading a class, instantiating a new instance of a class, determining class and instance methods, invoking class and instance methods, accessing and manipulating a class, fields, methods and constructors, determining the modifiers for fields, methods, classes, and interfaces, and/or other functionalities. The above described dynamic code, dynamic class loading, reflection, and/or other functionalities are similarly provided in the .NET platform through its tools such as, for example, System.CodeDom.Compiler namespace, System.Reflection.Emit namespace, and/or other native or other .NET tools. Other platforms in addition to Java and .NET may provide similar tools and/or functionalities. In some designs, dynamic code, dynamic class loading, reflection, and/or other functionalities can be used to facilitate modification of an application by inserting or injecting instruction sets (i.e. anticipatory Instruction Sets 526, etc.) into a running application. For example, an existing or dynamically created class comprising DCADO Unit 100 functionalities can be loaded into a running application through manual, automatic, or dynamic instrumentation. Once the class is created and loaded, an instance of DCADO Unit 100 class may be constructed. The instance of DCADO Unit 100 can then take or exert control of the application and/or implement alternate instruction sets (i.e. anticipatory Instruction Sets 526, etc.) at any point in the application's execution. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program 18 can be implemented through independent tools that can be utilized across different platforms. Such tools provide instrumentation and/or other capabilities on more than one platform or computing system and may facilitate application modification or insertion of instruction sets (i.e. anticipatory Instruction Sets 526, etc.). Examples of these tools include Pin, DynamoRIO, DynInst, Kprobes, KernInst, OpenPAT, DTrace, SystemTap, and/or others. In some aspects, Pin and/or any of its elements, methods, and/or techniques can be utilized for dynamic instrumentation. Pin can perform instrumentation by taking control of an application after it loads into memory. Pin may insert itself into the address space of an executing application enabling it to take control. Pin JIT compiler can then compile and implement alternate code (i.e. anticipatory Instruction Sets 526, etc.). Pin provides an extensive API for instrumentation at several abstraction levels. Pin supports two modes of instrumentation, JIT mode and probe mode. JIT mode uses a just-in-time compiler to insert instrumentation and recompile program code while probe mode uses code trampolines for instrumentation. Pin was designed for architecture and operating system independence. In other aspects, KernInst and/or any of its elements, methods, and/or techniques can be utilized for dynamic instrumentation. KernInst includes an instrumentation framework designed for dynamically inserting code into a running kernel of an operating system. KernInst implements probe-based dynamic instrumentation where code can be inserted, changed, and/or removed at will. Kerninst API enables client tools to construct their own tools for dynamic kernel instrumentation to suit variety of purposes such as insertion of alternate instruction sets (i.e. anticipatory Instruction Sets 526, etc.). Client tools can communicate with KernInst over a network (i.e. internet, wireless network, LAW, WAN, etc). Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program 18 can be implemented through utilizing operating system's native tools or capabilities such as Unix ptrace command. Ptrace includes a system call that may enable one process to control another allowing the controller to inspect and manipulate the internal state of its target. Ptrace can be used to modify a running application such as modify an application with alternate instruction sets (i.e. anticipatory Instruction Sets 526, etc.). By attaching to an application using the ptrace call, the controlling application can gain extensive control over the operation of its target. This may include manipulation of its instruction sets, execution path, file descriptors, memory, registers, and/or other components. Ptrace can single-step through the target's code, observe and intercept system calls and their results, manipulate the target's signal handlers, receive and send signals on the target's behalf, and/or perform other operations within the target application. Ptrace's ability to write into the target application's memory space enables the controller to modify the running code of the target application. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program 18 can be implemented through utilizing just-in-time (JIT) compiling. JIT compilation (also known as dynamic translation, dynamic compilation, etc.) includes compilation performed during an application's execution (i.e. runtime, etc.). A code can be compiled when it is about to be executed, and it may be cached and reused later without the need for additional compilation. In some aspects, a JIT compiler can convert source code or byte code into machine code. In other aspects, a JIT compiler can convert source code into byte code. JIT compiling may be performed directly in memory. For example, JIT compiler can output machine code directly into memory and immediately execute it.

Platforms such as Java, .NET, and/or others may implement JIT compilation as their native functionality. Platform independent tools for custom system design may include JIT compilation functionalities as well. In some aspects, JIT compilation includes redirecting application's execution to a JIT compiler from a specific entry point. For example, Pin can insert its JIT compiler into the address space of an application. Once execution is redirected to it, JIT compiler may receive alternate instruction sets (i.e. anticipatory Instruction Sets 526, etc.) immediately before their compilation. The JIT compiled instruction sets can be stored in memory or another repository from where they may be retrieved and executed. Alternatively, for example, JIT compiler can create a copy of the original application code or a segment thereof, and insert alternate code (i.e. anticipatory Instruction Sets 526, etc.) before compiling the modified code copy. In some aspects, JIT compiler may include a specialized memory such as fast cache memory dedicated to JIT compiler functionalities from which the modified code can be fetched rapidly. JIT compilation and/or any compilation in general may include compilation, interpretation, or other translation into machine code, bytecode, and/or other formats or types of code. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program 18 can be implemented through dynamic recompilation. Dynamic recompilation includes recompiling an application or part thereof during execution. An application can be modified with alternate features or instruction sets that may take effect after recompilation. Dynamic recompilation may be practical in various types of applications including object oriented, event driven, forms based, and/or other applications. In a typical windows-based application, most of the action after initial startup occurs in response to user or system events such as moving the mouse, selecting a menu option, typing text, running a scheduled task, making a network connection, and/or other events when an event handler is called to perform an operation appropriate for the event. Generally, when no events are being generated, the application is idle. For example, when an event occurs and an appropriate event handler is called, instrumentation can be implemented in the application's source code to insert alternate instruction sets (i.e. anticipatory Instruction Sets 526, etc.) at which point the modified source code can be recompiled and/or executed. In some aspects, the state of the application can be saved before recompiling its modified source code so that the application may continue from its prior state. Saving the application's state can be achieved by saving its variables, data structures, objects, location of its current instruction, and/or other necessary information in environmental variables, memory, or other repositories where they can be accessed once the application is recompiled. In other aspects, application's variables, data structures, objects, address of its current instruction, and/or other necessary information can be saved in a repository such as file, database, or other repository accessible to the application after recompilation of its source code. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program 18 can be implemented through modifying or redirecting Application Program's 18 execution path. Generally, an application can be loaded into memory and the flow of execution proceeds from one instruction set to the next until the end of the application. An application may include a branching mechanism that can be driven by keyboard or other input devices, system events, and/or other computing system components or events that may impact the execution path. The execution path can also be altered by an external application through acquiring control of execution and/or redirecting execution to a function, routine/subroutine, or an alternate code segment at any point in the application's

execution. A branch, jump, or other mechanism can be utilized to implement the redirected execution. For example, a jump instruction can be inserted at a specific point in an application's execution to redirect execution to an alternate code segment. A jump instruction set may include, for example, an unconditional branch, which always results in branching, or a conditional branch, which may or may not result in branching depending on a condition. When executing an application, a computer may fetch and execute instruction sets in sequence until it encounters a branch instruction set. If the instruction set is an unconditional branch, or it is conditional and the condition is satisfied, the computer may fetch its next instruction set from a different instruction set sequence or code segment as specified by the branch instruction set. After the execution of the alternate code segment, control may be redirected back to the original jump point or to another point in the application. For example, modifying an application can be implemented by redirecting execution of an application to alternate instruction sets (i.e. anticipatory Instruction Sets 526, etc.). Alternate instruction sets can be pre-compiled, pre-interpreted, or otherwise pre-translated and ready for execution. Alternate instruction sets can also be JIT compiled, JIT interpreted, or otherwise JIT translated before execution. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program 18 can be implemented through assembly language. Assembly language instructions may be directly related with the architecture's machine instructions as previously described. Assembly language can, therefore, be a powerful tool for implementing direct hardware (i.e. processor registers, memory, etc.) access and manipulations as well as access and manipulations of specialized processor features or instructions. Assembly language can also be a powerful tool for implementing low-level embedded systems, real-time systems, interrupt handlers, self or dynamically modifying code, and/or other applications. Specifically, for instance, self or dynamically modifying code that can be used to facilitate modifying of an application can be seamlessly implemented using assembly language. For example, using assembly language, instruction sets can be dynamically created and loaded into memory similar to the ones that a compiler may generate. Furthermore, using assembly language, memory space of a loaded application can be accessed to modify (including rewrite, overwrite, etc.) original instruction sets or to insert jumps or branches to alternate code elsewhere in memory. Some operating systems may implement protection from changes to applications loaded into memory. Operating system's, processor's, or other low level features or commands can be used to unprotect the protected locations in memory before the change as previously described. Alternatively, a pointer that may reside in a memory location where it could be readily altered can be utilized where the pointer may reference alternate code. In one example, assembly language can be utilized to write alternate code (i.e. anticipatory Instruction Sets 526, etc.) into a location in memory outside a running application's memory space. Assembly language can then be utilized to redirect the application's execution to the alternate code by inserting a jump or branch into the application's in-memory code, by redirecting program counter, or by other technique. In another example, assembly language can be utilized to overwrite or rewrite the entire or part of an application's in-memory code with alternate code. In some aspects, high-level programming languages can call an external assembly language program to facilitate application modification as previously described. In yet other aspects, relatively low-level programming languages such as C may allow embedding assembly language directly in their source code such as, for example, using asm keyword of C. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program 18 can be implemented through binary rewriting. Binary rewriting tools and/or techniques may modify an application's executable. In some aspects, modification can be minor such as in the case of optimization where the original executable's functionality is kept. In other aspects, modification may change the application's functionality such as by inserting alternate code (i.e. anticipatory Instruction Sets 526, etc.). Examples of binary rewriting tools include SecondWrite, ATOM, DynamoRIO, Purify, Pin, EEL, DynInst, PLTO, and/or others. Binary rewriting may include disassembly, analysis, and/or modification of target application. Since binary rewriting works directly on machine code executable, it is independent of source language, compiler, virtual machine (if one is utilized), and/or other higher level abstraction layers. Also, binary rewriting tools can perform application modifications without access to original source code. Binary rewriting tools include static rewriters, dynamic rewriters, minimally-invasive rewriters, and/or others. Static binary rewriters can modify an executable when the executable is not in use (i.e. not running). The rewritten executable may then be executed including any new or modified functionality. Dynamic binary rewriters can modify an executable during its execution, thereby enabling modification of an application's functionality at runtime. In some aspects, dynamic rewriters can be used for instrumentation or selective modifications such as insertion of alternate code (i.e. anticipatory Instruction Sets 526, etc.), and/or for other runtime transformations or modifications. For example, some dynamic rewriters can be configured to intercept an application's execution at indirect control transfers and insert instrumentation or other application modifying code. Minimally-invasive rewriters may keep the original machine code to the greatest extent possible. They support limited modifications such as insertion of jumps into and out of instrumented code. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

Referring to Fig. 30, in a further example, modifying execution and/or functionality of Processor 11 can be implemented through modification of processor registers, memory, or other computing system components. In some aspects, modifying execution and/or functionality of Processor 11 can be implemented by redirecting Processor's 11 execution to alternate instruction sets (i.e. anticipatory Instruction Sets 526, etc.). In one example, Program Counter 211 may hold or point to a memory address of the next instruction set that will be executed by Processor 11. Artificial Intelligence Unit 110 may generate anticipatory Instruction Sets 526 and store them in Memory 12 as previously described. Modification Interface 130 may then change Program Counter 211 to point to the location in Memory 12 where anticipatory Instruction Sets 526 are stored. The anticipatory Instruction Sets 526 can then be fetched from the location in Memory 12 pointed to by the modified Program Counter 211 and loaded into Instruction Register 212 for decoding and execution. Once anticipatory Instruction Sets 526 are executed, Modification Interface 130 may change Program Counter 211 to point to the last instruction set before the redirection or to any other instruction set. In other aspects, anticipatory Instruction Sets 526 can be loaded directly into Instruction Register 212. As previously described, examples of other processor or computing system components that can be used during an instruction cycle include memory address register (MAR), memory data register (MDR), data registers, address registers, general purpose registers (GPRs), conditional registers, floating point registers (FPRs), constant registers, special purpose registers, machine-specific registers, Register Array 214, Arithmetic Logic Unit 215, control unit, and/or other circuits or components. Any of the aforementioned processor registers, memory, or other computing system components can be accessed and/or modified to facilitate the disclosed functionalities. In some embodiments, processor interrupt may be issued to facilitate such access and/or modification. In some

designs, modifying execution and/or functionality of Processor 11 can be implemented in a program, combination of programs and hardware, or purely hardware system. Dedicated hardware may be built to perform modifying execution and/or functionality of Processor 11 with marginal or no impact to computing overhead. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

Referring to Figs. 31A-31B, in a further example, modifying execution and/or functionality of Logic Circuit 250 can be implemented through modification of inputs and/or outputs of Logic Circuit 250. While Processor 11 includes any type of logic circuit, Logic Circuit 250 is described separately herein to offer additional detail on its functioning. Logic Circuit 250 comprises the functionality for performing logic operations using the circuit's inputs and producing outputs based on the logic operations performed as previously described. In one example, Logic Circuit 250 may perform some logic operations using four input values and produce two output values. Modifying execution and/or functionality of Logic Circuit 250 can be implemented by replacing its input values with anticipatory input values (i.e. anticipatory Instruction Sets 526, etc.). Artificial Intelligence Unit 110 may generate anticipatory input values as previously described. Modification Interface 130 can then transmit the anticipatory input values to Logic Circuit 250 through the four hardwired connections as shown in Fig. 31A. Modification Interface 130 may use Switches 251 to prevent delivery of any input values that may be sent to Logic Circuit 250 from its usual input source. As such, DCADO Unit 100 may cause Logic Circuit 250 to perform its logic operations using the four anticipatory input values, thereby implementing autonomous Device 98 operation. In another example, Logic Circuit 250 may perform some logic operations using four input values and produce two output values. Modifying execution and/or functionality of Logic Circuit 250 can be implemented by replacing its output values with anticipatory output values (i.e. anticipatory Instruction Sets 526, etc.). Artificial Intelligence Unit 110 may generate anticipatory output values (i.e. anticipatory Instruction Sets 526, etc.) as previously described. Modification Interface 130 can then transmit the anticipatory output values through the two hardwired connections as shown in Fig. 31B. Modification Interface 130 may use Switches 251 to prevent delivery of any output values that may be sent by Logic Circuit 250. As such, DCADO Unit 100 may bypass Logic Circuit 250 and transmit the two anticipatory output values to downstream elements, thereby implementing autonomous Device 98 operation. In a further example, instead of or in addition to modifying input and/or output values of Logic Circuit 250, the execution and/or functionality of Logic Circuit 250 may be modified by modifying values or signals in one or more Logic Circuit's 250 internal components such as registers, memories, buses, and/or others (i.e. similar to the previously described modifying of Processor 11 components, etc.). In some designs, modifying execution and/or functionality of Logic Circuit 250 can be implemented in a program, combination of programs and hardware, or purely hardware system. Dedicated hardware may be built to perform modifying execution and/or functionality of Logic Circuit 250 with marginal or no impact to computing overhead. Any of the elements and/or techniques for modifying execution and/or functionality of Logic Circuit 250 can similarly be implemented with Processor 11 and/or other processing elements.

In some embodiments, DCADO Unit 100 may directly modify the functionality of an actuator (previously described, not shown). For example, Logic Circuit 250 or other processing element may control an actuator that enables Device 98 to perform mechanical, physical, and/or other operations. An actuator may receive one or more input values or control signals from Logic Circuit 250 or other processing element directing the actuator to perform specific operations. Modifying functionality of an actuator can be implemented by replacing its input values with anticipatory input values (i.e. anticipatory Instruction Sets 526, etc.) as previously described with respect to replacing

input values of Logic Circuit 250. Specifically, for instance, Artificial Intelligence Unit 110 may generate anticipatory input values as previously described. Modification Interface 130 can then transmit the anticipatory input values to the actuator. Modification Interface 130 may use Switches 251 to prevent delivery of any input values that may be sent to the actuator from its usual input source. As such, DCADO Unit 100 may cause the actuator to perform its operations using the anticipatory input values, thereby implementing autonomous Device 98 operation.

One of ordinary skill in art will recognize that Figs. 31A-31B depict one of many implementations of Logic Circuit 250 and that any number of input and/or output values can be utilized in alternate implementations. One of ordinary skill in art will also recognize that Logic Circuit 250 may include any number and/or combination of logic components to implement any logic operations.

Other additional techniques or elements can be utilized as needed for modifying execution and/or functionality of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing elements, or some of the disclosed techniques or elements can be excluded, or a combination thereof can be utilized in alternate embodiments.

Referring to Fig. 32, the illustration shows an embodiment of a method 9100 for learning and/or using a device's circumstances for autonomous device operation. In some aspects, the method can be used on a computing enabled device or system to enable learning of a device's operation in circumstances including objects with various properties and enable autonomous device operation in similar circumstances. Method 9100 may include any action or operation of any of the disclosed methods such as method 9200, 9300, 9400, 9500, 9600, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method 9100.

At step 9105, a first collection of object representations is received. A collection of object representations (i.e. Collection of Object Representations 525, etc.) may include one or more object representations (i.e. Object Representations 625, etc.), object properties (i.e. Object Properties 630, etc.), and/or other elements or information. An object representation may include an electronic representation of an object (i.e. Object 615, etc.) detected in a device's surrounding. In some aspects, a collection of object representations may include one or more object representations, object properties, and/or other elements or information detected in a device's (i.e. Device's 98, etc.) surrounding at a particular time. A collection of object representations may, therefore, include knowledge (i.e. unit of knowledge, etc.) of a device's circumstances including objects with various properties at a particular time. In some designs, a collection of object representations may include or be associated with a time stamp (not shown), order (not shown), or other time related information. In some embodiments, a collection of object representations may include or be substituted with a stream of collections of object representations, and vice versa. Therefore, the terms collection of object representations and stream of collections of object representations may be used interchangeably herein depending on context. A stream of collections of object representations may include one collection of object representations or a group, sequence, or other plurality of collections of object representations. In some aspects, a stream of collections of object representations may include one or more collections of object representations, and/or other elements or information detected in a device's surrounding over time. A stream of collections of object representations may, therefore, include knowledge (i.e. unit of knowledge, etc.) of a device's circumstances including objects with various properties over time. As circumstances including objects with various properties in a device's surrounding change (i.e. objects and/or their properties change, move, act, transform, etc.) over time, this

change may be captured in a stream of collections of object representations. In some designs, each collection of object representations in a stream may include or be associated with the aforementioned time stamp, order, or other time related information. Examples of objects include biological objects (i.e. persons, animals, vegetation, etc.), nature objects (i.e. rocks, bodies of water, etc.), manmade objects (i.e. buildings, streets, ground/aerial/aquatic vehicles, etc.), and/or others. In some aspects, any part of an object may be detected as an object itself. For instance, instead of or in addition to detecting a vehicle as an object, a wheel and/or other parts of the vehicle may be detected as objects. In general, an object may include any object or part thereof that can be detected. Examples of object properties include existence of an object, type of an object (i.e. person, cat, vehicle, building, street, tree, rock, etc.), identity of an object (i.e. name, identifier, etc.), distance of an object, bearing/angle of an object, location of an object (i.e. distance and bearing/angle from a known point, object coordinates, etc.), shape/size of an object (i.e. height, width, depth, computer model, point cloud, etc.), activity of an object (i.e. motion, gestures, etc.), and/or other properties of an object. In general, an object property may include any attribute of an object (i.e. existence of an object, type of an object, identity of an object, shape/size of an object, etc.), any relationship of an object with the device, other objects, or the environment (i.e. distance of an object, bearing/angle of an object, friend/foe relationship, etc.), and/or other information related to an object. Objects and/or their properties can be detected by one or more sensors (i.e. Sensors 92, etc.) and/or an object processing unit (i.e. Object Processing Unit 93, etc.). A sensor may obtain or detect information about its environment. As such, one or more sensors can be used to detect objects and/or their properties in a device's surrounding. In some designs, a sensor may be part of a device whose circumstances are being used for DCADO functionalities. In other designs, a sensor may be part of a remote device whose circumstances are being used for DCADO functionalities. Examples of a sensor include a camera (i.e. Camera 92a, etc.), a microphone (i.e. Microphone 92b, etc.), a lidar (i.e. Lidar 92c, etc.), a radar (i.e. Radar 92d, etc.), a sonar (i.e. Sonar 92e, etc.), and/or others. An object processing unit may process output from a sensor to obtain information of interest. As such, an object processing unit can be used to process output from a sensor to detect objects and/or their properties in a device's surrounding. In some aspects, an object processing unit may create or generate a collection of object representations. In other aspects, an object processing unit may create or generate a stream of collections of object representations. An object processing unit may include a picture recognizer (i.e. Picture Recognizer 94a, etc.), a sound recognizer (i.e. Sound Recognizer 94b, etc.), a lidar processing unit (i.e. Lidar Processing Unit 94c, etc.), a radar processing unit (i.e. Radar Processing Unit 94d, etc.), a sonar processing unit (i.e. Sonar Processing Unit 94e, etc.), and/or other elements or functionalities. In general, an object processing unit may include any signal processing element or technique known in art as applicable. In some implementations, an object processing unit and/or any of its elements or functionalities can be included in sensor and/or other elements. Receiving comprises any action or operation by or for a Collection of Object Representations 525, stream of Collections of Object Representations 525, Object Representation 625, Object Property 630, Sensor 92, Camera 92a, Microphone 92b, Lidar 92c, Radar 92d, Sonar 92e, Object Processing Unit 93, Picture Recognizer 94a, Sound Recognizer 94b, Lidar Processing Unit 94c, Radar Processing Unit 94d, Sonar Processing Unit 94e, and/or other disclosed elements.

At step 9110, a first one or more instruction sets for operating a device are received. In some embodiments, an instruction set (i.e. Instruction Set 526, etc.) may be used or executed by a processor (i.e. Processor 11, etc.) in operating a device. In other embodiments, an instruction set may be part of an application program (i.e. Application

Program 18, etc.) used in operating a device. For example, the application can run or execute on one or more processors or other processing elements. In further embodiments, an instruction set may be used or executed by a logic circuit (i.e. Logic Circuit 250, etc.) in operating a device. For example, such instruction set may be or include one or more inputs into or outputs from a logic circuit. In further embodiments, an instruction set may be used by an actuator in operating a device. For example, such instruction set may be or include one or more inputs into an actuator. Operating a device includes performing or causing any operations on/by/with the device. In some designs, an instruction set can be received from a processor, application program, logic circuit, and/or other processing element as the instruction set is being used or executed. In other aspects, an instruction set can be received from a processor, application program, logic circuit, and/or other processing element after the instruction set is used or executed. In further aspects, an instruction set can be received from a processor, application program, logic circuit, and/or other processing element before the instruction set has been used or executed. In further aspects, an instruction set can be received from a running processor, running application program, running logic circuit, and/or other running processing element. As such, an instruction set can be received at runtime. In other designs, an instruction set can be received from an actuator. In further designs, an instruction set can be received from memory (i.e. Memory 12, etc.), hard drive, or any other storage element or repository. In further designs, an instruction set can be received over a network such as Internet, local area network, wireless network, and/or other network. In further designs, an instruction set can be received by an interface (i.e. Acquisition Interface 120, etc.) configured to obtain instruction sets from a processor, application program, logic circuit, actuator, and/or other element. In general, an instruction set can be received by any element of the system. One or more instruction sets may temporally correspond to a collection of object representations. In some aspects, an instruction set that temporally corresponds to a collection of object representations includes an instruction set used or executed at the time of generating the collection of object representations. In other aspects, an instruction set that temporally corresponds to a collection of object representations includes an instruction set used or executed within a certain time period before and/or after generating the collection of object representations. Any time period can be utilized depending on implementation. In further aspects, an instruction set that temporally corresponds to a collection of object representations includes an instruction set used or executed from the time of generating the collection of object representations to the time of generating a next collection of object representations. In further aspects, an instruction set that temporally corresponds to a collection of object representations includes an instruction set used or executed from the time of generating a preceding collection of object representations to the time of generating the collection of object representations. Any other temporal relationship or correspondence between collections of object representations and correlated instruction sets can be implemented. In general, one or more instruction sets that temporally correspond to a collection of object representations enable structuring knowledge of a device's operation at or around the time of generating the collection of object representations. Such functionality enables spontaneous or seamless learning of a device's operation in circumstances including objects with various properties as the device is operated in real life situations. In some embodiments, an instruction set may include one or more commands, keywords, symbols (i.e. parentheses, brackets, commas, semicolons, etc.), instructions, operators (i.e. =, <, >, etc.), variables, values, objects, data structures, functions (i.e. Function1(), FIRST(), MIN(), SQRT(), etc.), parameters, states, signals, inputs, outputs, references thereto, and/or other components. In other embodiments, an instruction set may include source code, bytecode, intermediate code, compiled, interpreted, or otherwise translated code,

runtime code, assembly code, machine code, and/or any other computer code. In further embodiments, an instruction set may include one or more inputs into and/or outputs from a logic circuit. In further embodiments, an instruction set may include one or more inputs into an actuator. Receiving comprises any action or operation by or for an Acquisition Interface 120, Instruction Set 526, and/or other disclosed elements.

At step 9115, the first collection of object representations is correlated with the first one or more instruction sets for operating the device. In some aspects, individual collections of object representations can be correlated with one or more instruction sets. In other aspects, streams of collections of object representations can be correlated with one or more instruction sets. In further aspects, individual collections of object representations or streams of collections of object representations can be correlated with the aforementioned temporally corresponding instruction sets. In further aspects, a collection of object representations or stream of collections of object representations may not be correlated with any instruction sets. Correlating may include structuring or generating a knowledge cell (i.e. Knowledge Cell 800, etc.) and storing one or more collections of object representations correlated with any instruction sets into the knowledge cell. Therefore, a knowledge cell may include any data structure or arrangement that can facilitate such storing. A knowledge cell includes knowledge (i.e. unit of knowledge, etc.) of how a device operated in a circumstance including objects with various properties. In some designs, extra information (i.e. Extra Info 527, etc.) may optionally be used to facilitate enhanced comparisons or decision making in autonomous device operation where applicable. Therefore, any collection of object representations, instruction set, and/or other element may include or be correlated with extra information. Extra information may include any information useful in comparisons or decision making performed in autonomous device operation. Examples of extra information include time information, location information, computed information, contextual information, and/or other information. Correlating may be omitted where learning of a device's operation in circumstances including objects with various properties is not implemented. Correlating comprises any action or operation by or for a Knowledge Structuring Unit 520, Knowledge Cell 800, and/or other disclosed elements.

At step 9120, the first collection of object representations correlated with the first one or more instruction sets for operating the device are stored. A collection of object representations correlated with one or more instruction sets may be part of a stored plurality of collections of object representations correlated with one or more instruction sets. Collections of object representations correlated with any instruction sets can be stored in a memory unit or other repository. The aforementioned knowledge cells comprising collections of object representations correlated with any instruction sets can be used in/as neurons, nodes, vertices, or other elements in any of the data structures or arrangements (i.e. neural networks, graphs, sequences, collection of knowledge cells, etc.) used for storing the knowledge of a device's operation in circumstances including objects with various properties. Knowledge cells may be connected, interrelated, or interlinked into knowledge structures using statistical, artificial intelligence, machine learning, and/or other models or techniques. Such interconnected or interrelated knowledge cells can be used for enabling autonomous device operation. The interconnected or interrelated knowledge cells may be stored or organized into a knowledgebase (i.e. Knowledgebase 530, etc.). In some embodiments, knowledgebase may be or include a neural network (i.e. Neural Network 530a, etc.). In other embodiments, knowledgebase may be or include a graph (i.e. Graph 530b, etc.). In further embodiments, knowledgebase may be or include a collection of sequences (i.e. Collection of Sequences 530c, etc.). In further embodiments, knowledgebase may be or include a sequence (i.e. Sequence 533, etc.). In further embodiments, knowledgebase may be or include a collection of knowledge cells

(i.e. Collection of Knowledge Cells 530d, etc.). In general, knowledgebase may be or include any data structure or arrangement, and/or repository capable of storing the knowledge of a device's operation in circumstances including objects with various properties. Knowledgebase may also include or be substituted with various artificial intelligence methods, systems, and/or models for knowledge structuring, storing, and/or representation such as deep learning, supervised learning, unsupervised learning, neural networks (i.e. convolutional neural network, recurrent neural network, deep neural network, etc.), search-based, logic and/or fuzzy logic-based, optimization-based, tree/graph/other data structure-based, hierarchical, symbolic and/or sub-symbolic, evolutionary, genetic, multi-agent, deterministic, probabilistic, statistical, and/or other methods, systems, and/or models. Storing may be omitted where learning of a device's operation in circumstances including objects with various properties is not implemented. Storing comprises any action or operation by or for a Knowledgebase 530, Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, Knowledge Cell 800, Node 852, Layer 854, Connection 853, Similarity Comparison 125, and/or other disclosed elements.

At step 9125, a new collection of object representations is received. Step 9125 may include any action or operation described in Step 9105 as applicable.

At step 9130, the new collection of object representations is compared with the first collection of object representations. Comparing one collection of object representations with another collection of object representations may include comparing at least a portion of one collection of object representations with at least a portion of the other collection of object representations. In some embodiments, collections of object representations may be compared individually. In some aspects, comparing of individual collections of object representations may include comparing one or more object representations of one collection of object representations with one or more object representations of another collection of object representations. In other aspects, comparing of object representations may include comparing one or more object properties of one object representation with one or more object properties of another object representation. In some designs, one or more object properties in the same category (i.e. Category 635, etc.) can be compared. Comparing may include any techniques for comparing text, numbers, and/or other data. In further aspects, some object representations, object properties, and/or other elements of a collection of object representations can be omitted from comparison depending on implementation. In other embodiments, collections of object representations may be compared collectively as part of streams of collections of object representations. Collective comparing of collections of object representations may include any features, functionalities, and embodiments of the aforementioned individual comparing of collections of object representations. In some aspects, collective comparing of collections of object representations may include comparing one or more collections of object representations of one stream of collections of object representations with one or more collections of object representations of another stream of collections of object representations. In some designs, one or more corresponding (i.e. similarly ordered, temporally related, etc.) collections of object representations from the compared streams of collections of object representations can be compared. In other designs, Dynamic Time Warping (DTW) and/or other techniques can be utilized for comparison and/or aligning temporal sequences (i.e. streams of collections of object representations, etc.) that may vary in time or speed. In further aspects, some collections of object representations can be omitted from comparison depending on implementation. Any combination of the aforementioned and/or other elements or techniques can be utilized in alternate embodiments of the comparing. Comparing may be omitted where anticipating of a device's operation in

circumstances including objects with various properties is not implemented. Comparing comprises any action or operation by or for a Decision-making Unit 540, Similarity Comparison 125, and/or other disclosed elements.

At step 9135, a determination is made that there is at least a partial match between the new collection of object representations and the first collection of object representations. In some embodiments, determining at least a partial match between individually compared collections of object representations includes determining that a similarity between one or more portions of one collection of object representations and one or more portions of another collection of object representations exceeds a similarity threshold. In other embodiments, determining at least a partial match between individually compared collections of object representations includes determining at least a partial match between one or more portions of one collection of object representations and one or more portions of another collection of object representations. In further embodiments, determining at least a partial match between individually compared collections of object representations includes determining substantial similarity between one or more portions of one collection of object representations and one or more portions of another collection of object representations. A portion of a collection of object representations may include an object representation, an object property, and/or other portion or element of the collection of object representations. In further embodiments, determining at least a partial match between individually compared collections of object representations includes determining that the number or percentage of matching or substantially matching object representations of the compared collections of object representations exceeds a threshold number (i.e. 1, 2, 4, 7, 18, etc.) or threshold percentage (i.e. 41%, 62%, 79%, 85%, 93%, etc.). In some aspects, type of object representations, importance of object representations, and/or other elements or techniques relating to object representations can be utilized for determining similarity using object representations. In further aspects, some of the object representations can be omitted in determining similarity using object representations depending on implementation. In further embodiments, determining a match or substantial match between compared object representations includes determining that the number or percentage of matching or substantially matching object properties of the compared object representations exceeds a threshold number (i.e. 1, 2, 3, 6, 11, etc.) or a threshold percentage (i.e. 55%, 61%, 78%, 82%, 99%, etc.). In some aspects, categories of object properties, importance of object properties, and/or other elements or techniques relating to object properties can be utilized for determining similarity using object properties. In further aspects, some of the object properties can be omitted in determining similarity using object properties depending on implementation. In some designs, substantial similarity of individually compared collections of object representations can be achieved when a similarity between one or more portions of one collection of object representations and one or more portions of another collection of object representations exceeds a similarity threshold. In other designs, substantial similarity of individually compared collections of object representations can be achieved when the number or percentage of matching or substantially matching object representations of the compared collections of object representations exceeds a threshold number (i.e. 1, 2, 4, 7, 18, etc.) or threshold percentage (i.e. 41%, 62%, 79%, 85%, 93%, etc.). In further aspects, substantial similarity of compared object representations can be achieved when the number or percentage of matching or substantially matching object properties of the compared object representations exceeds a threshold number (i.e. 1, 2, 3, 6, 11, etc.) or a threshold percentage (i.e. 55%, 61%, 78%, 82%, 99%, etc.). In some embodiments, determining at least a partial match between collectively compared collections of object representations (i.e. streams of collections of object representations, etc.) includes determining that the number or percentage of matching or

substantially matching collections of object representations of the compared streams of collections of object representations exceeds a threshold number (i.e. 1, 2, 4, 9, 33, 138, etc.) or threshold percentage (i.e. 39%, 58%, 77%, 88%, 94%, etc.). In some aspects, importance of collections of object representations, order of collections of object representations, and/or other elements or techniques relating to collections of object representations can be utilized for determining similarity of collectively compared collections of object representations or streams of collections of object representations. In further aspects, some of the collections of object representations can be omitted in determining similarity of collectively compared collections of object representations or streams of collections of object representations depending on implementation. In some designs, a threshold for a number or percentage similarity can be used to determine a match or substantial match between any of the aforementioned elements. Any text, number, and/or other data similarity determination techniques can be used in any of the aforementioned similarity determinations. A partial match of any of the compared elements may include a substantially or otherwise similar match, and vice versa. Therefore, these terms may be used interchangeably herein depending on context. Although, substantial similarity or substantial match is frequently used herein, it should be understood that any level of similarity, however high or low, may be utilized as defined by the rules (i.e. thresholds, etc.) for similarity. Any combination of the aforementioned and/or other elements or techniques can be utilized in alternate embodiments. Determining may be omitted where anticipating of a device's operation in circumstances including objects with various properties is not implemented. Determining comprises any action or operation by or for a Decision-making Unit 540, Similarity Comparison 125, and/or other disclosed elements.

At step 9140, the first one or more instruction sets for operating the device correlated with the first collection of object representations are executed. Executing may be performed in response to the aforementioned determining. Executing may be caused by DCADO Unit 100, Artificial Intelligence Unit 110, Modification Interface 130, and/or other disclosed elements. An instruction set may be executed by a processor (i.e. Processor 11, etc.), application program (i.e. Application Program 18, etc.), logic circuit (i.e. Logic Circuit 250, etc.), and/or other processing element. An instruction set may be executed or acted upon by an actuator. In some aspects, instruction sets (i.e. the one or more instruction sets for operating the device correlated with the first collection of object representations, etc.) anticipated or determined to be used or executed in a device's autonomous operation may be referred to as anticipatory instruction sets, alternate instruction sets, and/or other suitable name or reference. Therefore, these terms can be used interchangeably herein depending on context. Executing may include executing one or more alternate instruction sets (i.e. anticipatory instruction sets, etc.) instead of or prior to an instruction set that would have been executed in a regular course of execution. In some embodiments, executing may include modifying a register or other element of a processor with one or more alternate instruction sets. Executing may also include redirecting a processor to one or more alternate instruction sets. In other embodiments, processor may be or comprises a logic circuit. Executing may further include modifying an element of a logic circuit with one or more alternate instruction sets, redirecting the logic circuit to one or more alternate instruction sets, replacing the inputs into the logic circuit with one or more alternate inputs or instruction sets, and/or replacing the outputs from the logic circuit with one or more alternate outputs or instruction sets. Executing may further include replacing the inputs into an actuator with one or more alternate inputs or instruction sets. In further embodiments, a processor may run an application including instruction sets for operating a device. In some aspects, executing includes executing one or more alternate instruction sets as part of the application. In other aspects, executing includes modifying the

application. In further aspects, executing includes redirecting the application to one or more alternate instruction sets. In further aspects, executing includes modifying one or more instruction sets of the application. In further aspects, executing includes modifying the application's source code, bytecode, intermediate code, compiled code, interpreted code, translated code, runtime code, assembly code, machine code, or other code. In further aspects, executing includes modifying memory, processor register, storage, repository or other element where the application's instruction sets are stored or used. In further aspects, executing includes modifying instruction sets used for operating an object of the application. In further aspects, executing includes modifying an element of a processor, an element of a device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input used in running the application. In further aspects, executing includes modifying the application at source code write time, compile time, interpretation time, translation time, linking time, loading time, runtime, or other time. In further aspects, executing includes modifying one or more of the application's lines of code, statements, instructions, functions, routines, subroutines, basic blocks, or other code segments. In further aspects, executing includes a manual, automatic, dynamic, just in time (JIT), or other instrumentation of the application. In further aspects, executing includes utilizing one or more of a .NET tool, .NET application programming interface (API), Java tool, Java API, operating system tool, independent tool, or other tool for modifying the application. In further aspects, executing includes utilizing a dynamic, interpreted, scripting, or other programming language. In further aspects, executing includes utilizing dynamic code, dynamic class loading, or reflection. In further aspects, executing includes utilizing assembly language. In further aspects, executing includes utilizing metaprogramming, self-modifying code, or an application modification tool. In further aspects, executing includes utilizing just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further aspects, executing includes utilizing dynamic expression creation, dynamic expression execution, dynamic function creation, or dynamic function execution. In further aspects, executing includes adding or inserting additional code into the application's code. In further aspects, executing includes modifying, removing, rewriting, or overwriting the application's code. In further aspects, executing includes branching, redirecting, extending, or hot swapping the application's code. Branching or redirecting an application's code may include inserting a branch, jump, or other means for redirecting the application's execution. Executing comprises any action or operation by or for a Processor 11, Application Program 18, Logic Circuit 250, Modification Interface 130, and/or other disclosed elements.

At step 9145, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations are performed by the device. The one or more operations may be performed in response to the aforementioned executing. In some aspects, an operation includes any operation that can be performed by/with/on a computing enabled device. In other aspects, an operation includes any operation that can be performed by/with/on an actuator. In further aspects, an operation includes any operation that can be performed by/with/on a computer. In general, an operation includes any operation that can be performed by/with/on a device or element thereof. One of ordinary skill in art will recognize that, while all possible variations of operations by/with/on a device are too voluminous to describe and limited only by the device's design and/or user's utilization, all operations are within the scope of this disclosure in various implementations.

Referring to Fig. 33, the illustration shows an embodiment of a method 9200 for learning and/or using a device's circumstances for autonomous device operation. In some aspects, the method can be used on a computing enabled device or system to enable learning of a device's operation in circumstances including objects with various

properties and enable autonomous device operation in similar circumstances. Method 9200 may include any action or operation of any of the disclosed methods such as method 9100, 9300, 9400, 9500, 9600, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method 9200.

At step 9205, a first collection of object representations is received. Step 9205 may include any action or operation described in Step 9105 of method 9100 as applicable.

At step 9210, a first one or more instruction sets for operating a device are received. Step 9210 may include any action or operation described in Step 9110 of method 9100 as applicable.

At step 9215, the first collection of object representations correlated with the first one or more instruction sets for operating the device are learned. Step 9215 may include any action or operation described in Step 9115 and/or Step 9120 of method 9100 as applicable.

At step 9220, a new collection of object representations is received. Step 9220 may include any action or operation described in Step 9125 of method 9100 as applicable.

At step 9225, the first one or more instruction sets for operating the device correlated with the first collection of object representations are anticipated based on at least a partial match between the new collection of object representations and the first collection of object representations. Step 9225 may include any action or operation described in Step 9130 and/or Step 9135 of method 9100 as applicable.

At step 9230, the first one or more instruction sets for operating the device correlated with the first collection of object representations are executed. Step 9230 may include any action or operation described in Step 9140 of method 9100 as applicable.

At step 9235, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations are performed by the device. Step 9235 may include any action or operation described in Step 9145 of method 9100 as applicable.

Referring to Fig. 34, the illustration shows an embodiment of a method 9300 for learning and/or using a device's circumstances for autonomous device operation. In some aspects, the method can be used on a computing enabled device or system to enable learning of a device's operation in circumstances including objects with various properties and enable autonomous device operation in similar circumstances. Method 9300 may include any action or operation of any of the disclosed methods such as method 9100, 9200, 9400, 9500, 9600, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method 9300.

At step 9305, a first stream of collections of object representations is received. Step 9305 may include any action or operation described in Step 9105 of method 9100 as applicable.

At step 9310, a first one or more instruction sets for operating a device are received. Step 9310 may include any action or operation described in Step 9110 of method 9100 as applicable.

At step 9315, the first stream of collections of object representations is correlated with the first one or more instruction sets for operating the device. Step 9315 may include any action or operation described in Step 9115 of method 9100 as applicable.

At step 9320, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device are stored. Step 9320 may include any action or operation described in Step 9120 of method 9100 as applicable.

At step 9325, a new stream of collections of object representations is received. Step 9325 may include any action or operation described in Step 9125 of method 9100 as applicable.

At step 9330, the new stream of collections of object representations is compared with the first stream of collections of object representations. Step 9330 may include any action or operation described in Step 9130 of method 9100 as applicable.

At step 9335, a determination is made that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. Step 9335 may include any action or operation described in Step 9135 of method 9100 as applicable.

At step 9340, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations are executed. Step 9340 may include any action or operation described in Step 9140 of method 9100 as applicable.

At step 9345, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations are performed by the device. Step 9345 may include any action or operation described in Step 9145 of method 9100 as applicable.

Referring to Fig. 35, the illustration shows an embodiment of a method 9400 for learning and/or using a device's circumstances for autonomous device operation. In some aspects, the method can be used on a computing enabled device or system to enable learning of a device's operation in circumstances including objects with various properties and enable autonomous device operation in similar circumstances. Method 9400 may include any action or operation of any of the disclosed methods such as method 9100, 9200, 9300, 9500, 9600, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method 9400.

At step 9405, a first collection of object representations is received. Step 9405 may include any action or operation described in Step 9105 of method 9100 as applicable.

At step 9410, a first one or more inputs are received, wherein the first one or more inputs are also received by a logic circuit, and wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device. Step 9410 may include any action or operation described in Step 9110 of method 9100 as applicable.

At step 9415, the first collection of object representations is correlated with the first one or more inputs. Step 9415 may include any action or operation described in Step 9115 of method 9100 as applicable.

At step 9420, the first collection of object representations correlated with the first one or more inputs are stored. Step 9420 may include any action or operation described in Step 9120 of method 9100 as applicable.

At step 9425, a new collection of object representations is received. Step 9425 may include any action or operation described in Step 9125 of method 9100 as applicable.

At step 9430, the new collection of object representations is compared with the first collection of object representations. Step 9430 may include any action or operation described in Step 9130 of method 9100 as applicable.

At step 9435, a determination is made that there is at least a partial match between the new collection of object representations and the first collection of object representations. Step 9435 may include any action or operation described in Step 9135 of method 9100 as applicable.

At step 9440, the first one or more inputs correlated with the first collection of object representations are received by the logic circuit. Step 9440 may include any action or operation described in Step 9140 of method 9100 as applicable.

At step 9445, one or more operations defined by one or more outputs for operating the device produced by the logic circuit are performed by the device. Step 9445 may include any action or operation described in Step 9145 of method 9100 as applicable.

Referring to Fig. 36, the illustration shows an embodiment of a method 9500 for learning and/or using a device's circumstances for autonomous device operation. In some aspects, the method can be used on a computing enabled device or system to enable learning of a device's operation in circumstances including objects with various properties and enable autonomous device operation in similar circumstances. Method 9500 may include any action or operation of any of the disclosed methods such as method 9100, 9200, 9300, 9400, 9600, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method 9500.

At step 9505, a first collection of object representations is received. Step 9505 may include any action or operation described in Step 9105 of method 9100 as applicable.

At step 9510, a first one or more outputs are received, the first one or more outputs transmitted from a logic circuit, wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device. Step 9510 may include any action or operation described in Step 9110 of method 9100 as applicable.

At step 9515, the first collection of object representations is correlated with the first one or more outputs. Step 9515 may include any action or operation described in Step 9115 of method 9100 as applicable.

At step 9520, the first collection of object representations correlated with the first one or more outputs are stored. Step 9520 may include any action or operation described in Step 9120 of method 9100 as applicable.

At step 9525, a new collection of object representations is received. Step 9525 may include any action or operation described in Step 9125 of method 9100 as applicable.

At step 9530, the new collection of object representations is compared with the first collection of object representations. Step 9530 may include any action or operation described in Step 9130 of method 9100 as applicable.

At step 9535, a determination is made that there is at least a partial match between the new collection of object representations and the first collection of object representations. Step 9535 may include any action or operation described in Step 9135 of method 9100 as applicable.

At step 9540, one or more operations defined by the first one or more outputs correlated with the first collection of object representations are performed by the device. Step 9540 may include any action or operation described in Step 9145 of method 9100 as applicable.

Referring to Fig. 37, the illustration shows an embodiment of a method 9600 for learning and/or using a device's circumstances for autonomous device operation. In some aspects, the method can be used on a computing

enabled device or system to enable learning of a device's operation in circumstances including objects with various properties and enable autonomous device operation in similar circumstances. Method 9600 may include any action or operation of any of the disclosed methods such as method 9100, 9200, 9300, 9400, 9500, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method 9600.

At step 9605, a first collection of object representations is received. Step 9605 may include any action or operation described in Step 9105 of method 9100 as applicable.

At step 9610, a first one or more inputs are received, wherein the first one or more inputs are also received by an actuator, and wherein the actuator is configured to receive inputs and perform motions. Step 9610 may include any action or operation described in Step 9110 of method 9100 as applicable.

At step 9615, the first collection of object representations is correlated with the first one or more inputs. Step 9615 may include any action or operation described in Step 9115 of method 9100 as applicable.

At step 9620, the first collection of object representations correlated with the first one or more inputs are stored. Step 9620 may include any action or operation described in Step 9120 of method 9100 as applicable.

At step 9625, a new collection of object representations is received. Step 9625 may include any action or operation described in Step 9125 of method 9100 as applicable.

At step 9630, the new collection of object representations is compared with the first collection of object representations. Step 9630 may include any action or operation described in Step 9130 of method 9100 as applicable.

At step 9635, a determination is made that there is at least a partial match between the new collection of object representations and the first collection of object representations. Step 9635 may include any action or operation described in Step 9135 of method 9100 as applicable.

At step 9640, the first one or more inputs correlated with the first collection of object representations are received by the actuator. Step 9640 may include any action or operation described in Step 9140 of method 9100 as applicable.

At step 9645, one or more motions defined by the first one or more inputs correlated with the first collection of object representations are performed by the actuator. Step 9645 may include any action or operation described in Step 9145 of method 9100 as applicable.

Referring to Fig. 38, in some exemplary embodiments, Device 98 may be or include Loader 98a. Loader 98a may be operated by User 50 in person or remotely. Loader 98a may include or be coupled to one or more Sensors 92 (i.e. collectively referred to as Sensor 92, etc.) such as Camera 92a, Microphone 92b, Lidar 92c, Radar 92d, Sonar 92e, etc. and/or Object Processing Unit 93 that can detect Objects 615aa-615ad, and/or other elements or information in Loader's 98a surrounding. Object Processing Unit 93 may include Picture Recognizer 94a, Sound Recognizer 94b, Lidar Processing Unit 94c, Radar Processing Unit 94d, Sonar Processing Unit 94e, and/or other elements or functionalities as applicable. Object Processing Unit 93 may create or generate one or more (i.e. stream, etc.) Collections of Object Representations 525 comprising Object Representations 625, Object Properties 630, and/or other elements or information representing Objects 615 detected in Loader's 98a surrounding. Loader 98a may also include or be controlled by Logic Circuit 250 (i.e. microcontroller, etc.), Processor 11 (i.e. including any Application Program 18 running thereon, etc.), and/or other processing element that receives User's 50 (i.e.

operator's, etc.) operating directions and causes desired operations with Loader 98a such as moving, maneuvering, collecting, lifting, unloading, and/or others. User 50 can interact with Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element through inputting operating directions via Human-machine Interface 23 such as one or more steering wheels, levers, pedals, buttons, or other input devices. For instance, responsive to User's 50 manipulating a steering wheel and one or more levers, Logic Circuit 250 or Processor 11 may cause Loader's 98a arm with bucket to collect a load, one or more motors or other actuators to move or maneuver Loader 98a, lifting system (i.e. hydraulic, pneumatic, mechanical, electrical, etc.) to lift a load, and/or arm with bucket to unload a load. Loader 98a may also include or be coupled to DCADO Unit 100. DCADO Unit 100 may be embedded (i.e. integrated, etc.) into or coupled to Loader's 98a Logic Circuit 250, Processor 11, and/or other processing element. DCADO Unit 100 may also be a program embedded (i.e. integrated, etc.) into or interfaced with Application Program 18 running on Processor 11 and/or other processing element. DCADO Unit 100 can obtain Instruction Sets 526 used or executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element. In some aspects, Instruction Sets 526 may include one or more inputs into or outputs from Loader's 98a Logic Circuit 250 (i.e. microcontroller, etc.). In other aspects, Instruction Sets 526 may include one or more instruction sets from Loader's 98a Processor's 11 registers or other components. In further aspects, Instruction Sets 526 may include one or more instruction sets used or executed in Application Program 18. DCADO Unit 100 may also optionally obtain any Extra Info 527 (i.e. time, location, computed, contextual, and/or other information, etc.) related to Loader's 98a operation. As User 50 operates Loader 98a in circumstances including objects with various properties as shown, DCADO Unit 100 may learn Loader's 98a operations in these circumstances by correlating Collections of Object Representations 525 representing Objects 615 detected in Loader's 98a surrounding with one or more Instruction Sets 526 used or executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element. Any Extra Info 527 related to Loader's 98a operation may also optionally be correlated with Collections of Object Representations 525. DCADO Unit 100 may store this knowledge into Knowledgebase 530 (i.e. Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.). In the future, DCADO Unit 110 may compare incoming Collections of Object Representations 525 representing Objects 615 detected in Loader's 98a surrounding with previously learned Collections of Object Representations 525 including optionally using any Extra Info 527 for enhanced decision making. If substantially similar or at least a partial match is found or determined, the Instruction Sets 526 correlated with the previously learned Collections of Object Representations 525 can be autonomously executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element, thereby enabling autonomous operation of Loader 98a in similar circumstances as in previously learned ones. For instance, Loader 98a comprising DCADO Unit 100 may learn User 50-directed collecting, moving, maneuvering, lifting, unloading, and/or other operations in a circumstance that includes Rock 615aa, Pile of Material 615ab, Person 615ac, Truck 615ad, and/or other Objects 615 among which Loader 98a may need to maneuver and/or with which Loader 98a may need to interact. In the future, when a circumstance that includes Objects 615 with similar Object Properties 630 is encountered, Loader 98a may implement collecting, moving, maneuvering, lifting, and/or unloading operations autonomously.

In some embodiments, DCADO Unit 100 may reside on Server 96 accessible over Network 95 as previously described. In such embodiments, any number of Loaders 98a may connect to such remote DCADO Unit

100 and the remote DCADO Unit 100 may learn their operations in circumstances including objects with various properties. In turn, any number of Loaders 98a can utilize the remote DCADO Unit 100 for autonomous operation in circumstances including objects with various properties. For example, multiple operators (i.e. Users 50, etc.) may operate their Loaders 98a that are configured to transmit their operations in circumstances including objects with various properties to a remote DCADO Unit 100. Such remote DCADO Unit 100 enables learning of the operators' collective knowledge of operating Loaders 98 in circumstances including objects with various properties. Any number of Loaders 98 can utilize such collective knowledge comprised in the remote DCADO Unit 100 for their autonomous operation. Any of the disclosed elements such as Artificial Intelligence Unit 110, Knowledgebase 530, and/or others may reside on Server 96, and any combination of local and remote elements can be implemented in alternate embodiments.

In some embodiments, Loader 98a may include or be coupled to a plurality of Sensors 92 and/or their corresponding Object Processing Units 93. In one example, multiple Sensors 92 may detect objects and/or their properties from different angles or on different sides of Loader 98a. In another example, one or more Sensors 92 may be placed on different sub-devices, sub-systems, or elements of Loader 98a. For instance, one Sensor 92 may be placed on the roof of Loader 98a, another Sensor 92 may be placed on the arm of Loader 98a, and an additional Sensor 92 may be placed on the bucket of Loader 98a. In some designs where multiple Sensors 92 are placed on different sub-devices, sub-systems, or elements of Loader 98a, multiple DCADO Units 100 can be utilized (i.e. one DCADO Unit 100 for each Sensor 92 or group of Sensors 92 and/or their corresponding Object Processing Units 93, etc.). In such designs, as User 50 operates Loader 98a in circumstances including objects with various properties, a particular DCADO Unit 100 may learn operations of Loader's 98a sub-device, sub-system, or element in these circumstances by correlating Collections of Object Representations 525 representing Objects 615 detected by Sensor 92 on the sub-device, sub-system, or element assigned to the DCADO Unit 100 with one or more Instruction Sets 526 used or executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element. The learning and/or decision making in Loader's 98a operation can, therefore, be performed per individual sub-device, sub-system, or element. In other designs where multiple Sensors 92 are placed on different sub-devices, sub-systems, or elements of Loader 98a, as User 50 operates Loader 98a in circumstances including objects with various properties, a single DCADO Unit 100 may learn Loader's 98a operations in these circumstances by correlating collective Collections of Object Representations 525 representing Objects 615 detected by Sensors 92 on the sub-devices, sub-systems, or elements with one or more Instruction Sets 526 used or executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element.

In some embodiments, Loader 98a may include a plurality of Logic Circuits 250, Processors 11, Application Programs 18, and/or other processing elements. In some aspects, one or more sub-devices, sub-systems, or elements of Loader 98a may be controlled by different processing elements. For example, one Processor 11 (i.e. including any Application Programs 18 running thereon, etc.) may control the moving system (i.e. drivetrain, powertrain, etc.) of Loader 98a, one Logic Circuit 250 may control an arm of Loader 98a, and an additional Logic Circuit 250 may control a bucket of Loader 98a. In some designs where multiple processing elements are utilized, multiple DCADO Units 100 can also be utilized (i.e. one DCADO Unit 100 for each processing element, etc.). In such designs, as User 50 operates Loader 98a in circumstances including objects with various properties, a particular DCADO Unit 100 may learn Loader's 98a operations in these circumstances by correlating Collections of Object

Representations 525 with one or more Instruction Sets 526 used or executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element assigned to the DCADO Unit 100. The learning and/or decision making in Loader's 98a operation can, therefore, be performed per individual processing element. In other designs where multiple processing elements are utilized, as User 50 operates Loader 98a in circumstances including objects with various properties, a single DCADO Unit 100 may learn Loader's 98a operations in these circumstances by correlating Collections of Object Representations 525 with collective Instruction Sets 526 used or executed by a plurality of Logic Circuits 250, Processors 11, Application Programs 18, and/or other processing elements.

In some embodiments, a combination of DCADO Unit 100 and other systems and/or techniques can be utilized to implement Loader's 98a operation. In one example, DCADO Unit 100 may be a primary or preferred system for implementing Loader's 98a operation. While operating autonomously under the control of DCADO Unit 100, Loader 98a may encounter a circumstance including objects with various properties that has not been encountered or learned before. In such situations, User 50 and/or non-DCADO system may take control of Loader's 98a operation. DCADO Unit 100 may take control again when Loader 98a encounters a previously learned circumstance including objects with various properties. Naturally, DCADO Unit 100 can learn Loader's 98a operation in the circumstances while User 50 and/or non-DCADO system is in control of Loader 98a, thereby reducing or eliminating the need for future involvement of User 50 and/or non-DCADO system. For instance, one User 50 can control or assist in controlling multiple Loaders 98a comprising DCADO Units 100. In such instances, User 50 can control or assist in controlling a Loader 98a that may encounter a circumstance including objects with various properties that has not been encountered or learned before while the Loaders 98a operating in previously learned circumstances can operate autonomously. In another example, User 50 and/or non-DCADO system may be a primary or preferred system for implementing Loader's 98a operation. While operating under the control of User 50 and/or non-DCADO system, User 50 and/or non-DCADO system may release control to DCADO Unit 100 for any reason (i.e. User 50 gets tired or distracted, non-DCADO system gets stuck or cannot make a decision, etc.), at which point Loader 98a can be controlled by DCADO Unit 100. In some designs, DCADO Unit 100 may take control in certain special circumstances including objects with various properties where DCADO Unit 100 may offer superior performance even though User 50 and/or non-DCADO system may generally be preferred. Once Loader 98a leaves such special circumstances, DCADO Unit 100 may release control to User 50 and/or non-DCADO system. In general, DCADO Unit 100 can take control from, share control with, or release control to User 50, non-DCADO system, and/or other system or process at any time, in any circumstances, and remain in control for any period of time as needed.

In some embodiments, DCADO Unit 100 may control one or more sub-devices, sub-systems, or elements of Loader 98a while User 50 and/or non-DCADO system may control other one or more sub-devices, sub-systems, or elements of Loader 98a. For example, User 50 and/or non-DCADO system may control the moving system (i.e. drivetrain, powertrain, etc.) of Loader 98a, while DCADO Unit 100 may control an arm and bucket of Loader 98a. Any other combination of controlling various sub-devices, sub-systems, or elements of Loader 98a by DCADO Unit 100 and User 50 and/or non-DCADO system can be implemented.

Referring to Fig. 39, in some exemplary embodiments, Device 98 may be or include Boat 98b. Boat 98b may be operated by User 50 in person or remotely. Boat 98b may include or be coupled to one or more Sensors 92

(i.e. collectively referred to as Sensor 92, etc.) such as Camera 92a, Microphone 92b, Lidar 92c, Radar 92d, Sonar 92e, etc. and/or Object Processing Unit 93 that can detect Objects 615ba-615bd, and/or other elements or information in Boat's 98b surrounding. Object Processing Unit 93 may include Picture Recognizer 94a, Sound Recognizer 94b, Lidar Processing Unit 94c, Radar Processing Unit 94d, Sonar Processing Unit 94e, and/or other elements or functionalities as applicable. Object Processing Unit 93 may create or generate one or more (i.e. stream, etc.) Collections of Object Representations 525 comprising Object Representations 625, Object Properties 630, and/or other elements or information representing Objects 615 detected in Boat's 98b surrounding. Boat 98b may also include or be controlled by Logic Circuit 250 (i.e. microcontroller, etc.), Processor 11 (i.e. including any Application Program 18 running thereon, etc.), and/or other processing element that receives User's 50 (i.e. operator's, etc.) operating directions and causes desired operations with Boat 98b such as moving, maneuvering, and/or other operations. User 50 can interact with Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element through inputting operating directions via Human-machine Interface 23 such as one or more steering wheels, levers, pedals, buttons, or other input devices. For instance, responsive to User's 50 manipulating a steering wheel and one or more levers, Logic Circuit 250 or Processor 11 may cause one or more motors or other actuators to move or maneuver Boat 98b. Boat 98b may also include or be coupled to DCADO Unit 100. DCADO Unit 100 may be embedded (i.e. integrated, etc.) into or coupled to Boat's 98b Logic Circuit 250, Processor 11, and/or other processing element. DCADO Unit 100 may also be a program embedded (i.e. integrated, etc.) into or interfaced with Application Program 18 running on Processor 11 and/or other processing element. DCADO Unit 100 can obtain Instruction Sets 526 used or executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element. In some aspects, Instruction Sets 526 may include one or more inputs into or outputs from Boat's 98b Logic Circuit 250 (i.e. microcontroller, etc.). In other aspects, Instruction Sets 526 may include one or more instruction sets from Boat's 98b Processor's 11 registers or other components. In further aspects, Instruction Sets 526 may include one or more instruction sets used or executed in Application Program 18. DCADO Unit 100 may also optionally obtain any Extra Info 527 (i.e. time, location, computed, contextual, and/or other information, etc.) related to Boat's 98b operation. As User 50 operates Boat 98b in circumstances including objects with various properties as shown, DCADO Unit 100 may learn Boat's 98b operations in these circumstances by correlating Collections of Object Representations 525 representing Objects 615 detected in Boat's 98b surrounding with one or more Instruction Sets 526 used or executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element. Any Extra Info 527 related to Boat's 98b operation may also optionally be correlated with Collections of Object Representations 525. DCADO Unit 100 may store this knowledge into Knowledgebase 530 (i.e. Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.). In the future, DCADO Unit 110 may compare incoming Collections of Object Representations 525 representing Objects 615 detected in Boat's 98b surrounding with previously learned Collections of Object Representations 525 including optionally using any Extra Info 527 for enhanced decision making. If substantially similar or at least a partial match is found or determined, the Instruction Sets 526 correlated with the previously learned Collections of Object Representations 525 can be autonomously executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element, thereby enabling autonomous operation of Boat 98b in similar circumstances as in previously learned ones. For instance, Boat 98b comprising DCADO Unit 100 may learn User 50-directed moving, maneuvering, and/or other operations in

a circumstance that includes Fishing Boat 615ba, Lighthouse 615bb, Sailboat 615bc, Cruise Ship 615bd, and/or other Objects 615 among which Boat 98b may need to maneuver. In the future, when a circumstance that includes Objects 615 with similar Object Properties 630 is encountered, Boat 98b may implement moving, maneuvering, and/or other operations autonomously. In some aspects, the shore (not enumerated) or any part thereof (i.e. cliff, ridge, beach, etc.) may be detected as an Object 615 itself, which may then be learned and used in autonomous operation of Boat 98b.

Referring to Fig. 40, in some exemplary embodiments, an Area of Interest 450 can be utilized. In one example, Area of Interest 450 may include a radial, circular, elliptical, or other such area around Boat 98b. In another example, Area of Interest 450 may include a triangular, rectangular, octagonal, or other such area around Boat 98b. In a further example, Area of Interest 450 may include a spherical, cubical, pyramid-like, or other such area around Boat 98b as applicable to 3D space. Any other Area of Interest 450 shape can be utilized depending on implementation. The shape and/or size of Area of Interest 450 can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. Utilizing Area of Interest 450 enables DCADO Unit 100 to focus on Boat's 98b immediate surrounding, thereby avoiding extraneous detail in the rest of the surrounding. In some aspects, Area of Interest 450 can be subdivided into sub-areas (i.e. sub-circles, sub-rectangles, sub-spheres, etc.). Sub-areas can be used to classify the surrounding by distance from Boat 98b. For example, the surrounding closer to Boat 98b may be more important and may be assigned higher importance index or weight. As User 50 operates Boat 98b in circumstances including objects with various properties as shown, DCADO Unit 100 may learn Boat's 98b operations in these circumstances by correlating Collections of Object Representations 525 representing Objects 615 detected in Area of Interest 450 around Boat 98b with one or more Instruction Sets 526 used or executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element. Any Extra Info 527 related to Boat's 98b operation may also optionally be correlated with Collections of Object Representations 525. DCADO Unit 100 may store this knowledge into Knowledgebase 530 (i.e. Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.). In the future, DCADO Unit 110 may compare incoming Collections of Object Representations 525 representing Objects 615 detected in Area of Interest 450 around Boat 98b with previously learned Collections of Object Representations 525 including optionally using any Extra Info 527 for enhanced decision making. If substantially similar or at least a partial match is found or determined, the Instruction Sets 526 correlated with the previously learned Collections of Object Representations 525 can be autonomously executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element, thereby enabling autonomous operation of Boat 98b in similar Areas of Interest 450 as in previously learned ones. For instance, Boat 98b comprising DCADO Unit 100 may learn User 50-directed moving, maneuvering, and/or other operations in an Area of Interest 450 that includes Fishing Boat 615ba, Lighthouse 615bb, Cruise Ship 615bd, and/or other Objects 615 among which Boat 98b may need to maneuver. In the future, when an Area of Interest 450 that includes Objects 615 with similar Object Properties 630 is encountered, Boat 98b may implement moving, maneuvering, and/or other operations autonomously.

The features, functionalities, and embodiments described with respect to Loader 98a and Boat 98b can be implemented in any situation where Device 98 may need to autonomously maneuver among, interact with, or perform other operations relative to objects in its surrounding. Therefore, the features, functionalities, and

embodiments described with respect to Loader 98a and Boat 98b can similarly be implemented on any computing enabled machine such as a bulldozer, an excavator, a crane, a forklift, a truck, a construction machine, an assembly machine, an object handling machine, an object dispensing machine, a sorting machine, a restocking machine, an industrial machine, an agricultural machine, a harvesting machine, a building control system, a home or other appliance, a toy, a robot, a tank, an aircraft, a vessel, a submarine, a ground vehicle, an aerial vehicle, an aquatic vehicle, and/or other computing-enabled machine or system.

In yet some exemplary embodiments, Device 98 may be or include a control device such as a thermostat, control panel, remote or other controller, and/or other control device. For instance, a thermostat comprising DCADO Unit 100 may learn User's 50 setting temperature of an air conditioning system controlled by the thermostat in a circumstance that includes User 50 and/or other persons entering or being present in a room. In the future, when a circumstance that includes User 50 and/or other persons entering or being present in the room is encountered, thermostat may implement setting temperature of the air conditioning system autonomously. In some aspects, a control device may be included in the device being controlled (i.e. control panel of an oven, refrigerator, fixture, etc.). In other aspects, a control device may be separate from the device being controlled (i.e. remote controller of a television device, etc.). In yet further exemplary embodiments, Device 98 may be or include a mobile computer such as a smartphone, tablet, and/or other mobile computer. For instance, a smartphone comprising DCADO Unit 100 may learn User 50-directed playing a music file, setting a vibrate mode, and/or other operations in a circumstance that includes objects with various properties. In the future, when a circumstance that includes objects with similar properties is encountered, smartphone may implement playing music file, setting vibrate mode, and/or other operations autonomously. In general, Device 98 may be or include any movable, stationary, or other device. One of ordinary skill in art will understand that Device 98 may be or include any device that can implement and/or benefit from the functionalities described herein.

It must be noted that as used herein and in the appended claims, the singular forms "a", "an", and "the" include plural referents unless the context clearly dictates otherwise.

A number of embodiments have been described herein. While this disclosure contains many specific implementation details, these should not be construed as limitations on the scope of the disclosure or of what may be claimed, but rather as descriptions of features specific to particular embodiments. It should be understood that various modifications can be made without departing from the spirit and scope of the disclosure. The logic flows depicted in the figures do not require the particular order shown, or sequential order, to achieve desirable results. In addition, other or additional steps, elements, or connections can be included, or some of the steps, elements, or connections can be eliminated, or a combination thereof can be utilized in the described flows, illustrations, or descriptions. Further, the various aspects of the disclosed devices, apparatuses, systems, and/or methods can be combined in whole or in part with each other to produce additional implementations. Moreover, separation of various components in the embodiments described herein should not be understood as requiring such separation in all embodiments, and it should be understood that the described components can generally be integrated together in a single product or packaged into multiple products. Accordingly, other embodiments are within the scope of the following claims.

CLAIMS

Claim 1. A system for learning and using a device's circumstances for autonomous device operating, the system implemented at least in part on one or more computing devices, the system comprising:

a processor circuit configured to execute instruction sets for operating a device;

a memory unit configured to store data;

a sensor configured to detect objects; and

an artificial intelligence unit configured to:

receive a first stream of collections of object representations, the first stream of collections of object representations including one or more collections of representations of objects detected by the sensor;

receive a first one or more instruction sets for operating the device;

learn the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device;

receive a new stream of collections of object representations, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor;

anticipate the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations; and

cause the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations performed in response to the executing by the processor circuit.

Claim 2. The system of Claim 1, wherein the receiving the first one or more instruction sets for operating the device includes a tracing of the processor circuit or a tracing of a component of the processor circuit.

Claim 3. The system of Claim 1, further comprising:

an application including the instruction sets for operating the device, the application running on the processor circuit, wherein the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from the application.

Claim 4. The system of Claim 3, wherein the receiving the first one or more instruction sets for operating the device from the application includes at least one of: a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of the application.

Claim 5. The system of Claim 3, wherein the receiving the first one or more instruction sets for operating the device from the application includes a tracing of the application.

Claim 6. The system of Claim 1, wherein the first one or more instruction sets for operating the device include one or more instruction sets that temporally correspond to the first stream of collections of object representations.

Claim 7. The system of Claim 1, wherein the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device include a knowledge of how the device operated in a circumstance.

Claim 8. The system of Claim 1, wherein the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device includes storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device into the memory unit, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device being part of a plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device stored in the memory unit.

Claim 9. The system of Claim 1, wherein at least one of: the processor circuit, the memory unit, the sensor, or the artificial intelligence unit are part of the device.

Claim 10. The system of Claim 1, wherein the artificial intelligence unit is part of, operating on, or coupled to the processor circuit.

Claim 11. A non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising:

receiving a first stream of collections of object representations, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor;

receiving a first one or more instruction sets for operating a device;

learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device;

receiving a new stream of collections of object representations, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor;

anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on

at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations; and

causing an execution of the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the execution.

Claim 12. The non-transitory computer storage medium of Claim 11, wherein the execution of the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations is performed by the one or more processor circuits or by another one or more processor circuits.

Claim 13. The non-transitory computer storage medium of Claim 11, wherein the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from an application, the application comprising instruction sets for operating the device.

Claim 14. The non-transitory computer storage medium of Claim 13, wherein the receiving the first one or more instruction sets for operating the device from the application includes a tracing of the application.

Claim 15. The non-transitory computer storage medium of Claim 11, wherein the first one or more instruction sets for operating the device include one or more instruction sets that temporally correspond to the first stream of collections of object representations.

Claim 16. A method comprising:

(a) receiving a first stream of collections of object representations by a processor circuit, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor;

(b) receiving a first one or more instruction sets for operating a device by the processor circuit;

(c) learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the processor circuit;

(d) receiving a new stream of collections of object representations by the processor circuit, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor;

(e) anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, the anticipating of (e) performed by the processor circuit;

(f) executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the executing of (f) performed in response to the anticipating of (e); and

(g) performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the executing of (f).

Claim 17. The method of Claim 16, wherein the executing of (f) is performed by the processor circuit or by another processor circuit.

Claim 18. The method of Claim 16, wherein the receiving of (b) includes receiving the first one or more instruction sets for operating the device from an application, the application comprising instruction sets for operating the device.

Claim 19. The method of Claim 18, wherein the receiving the first one or more instruction sets for operating the device from the application includes a tracing of the application.

Claim 20. The method of Claim 16, wherein the first one or more instruction sets for operating the device include one or more instruction sets that temporally correspond to the first stream of collections of object representations.

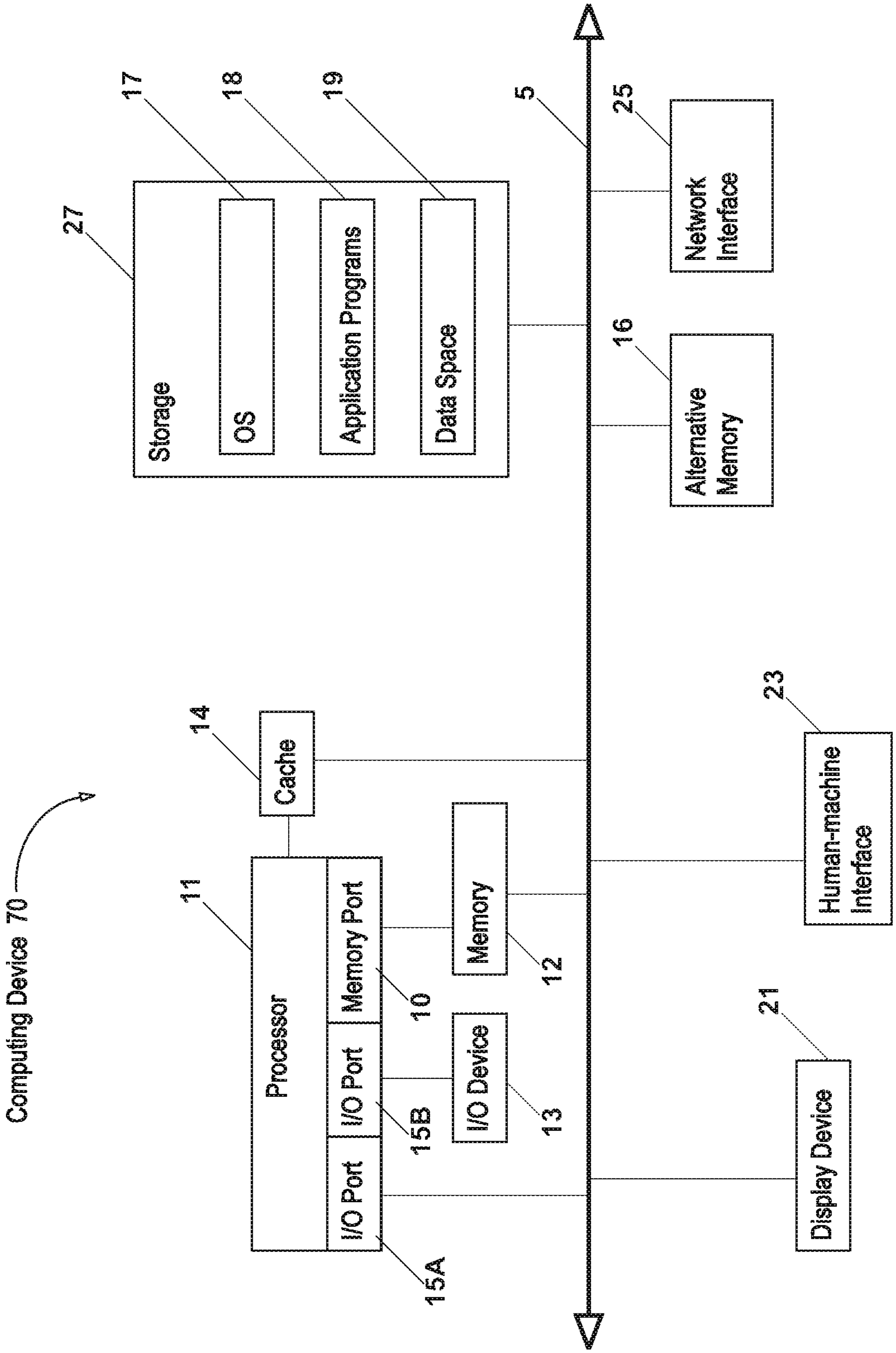

Computing Device 70
Storage
OS
Application Programs
Data Space
Network Interface
Alternative Memory
Cache
Processor
Memory Port
I/O Port
I/O Port
Memory
I/O Device
Human-machine Interface
Display Device
27
17
18
19
5
25
16
14
11
10
12
13
15B
15A
23
21

FIG. 1

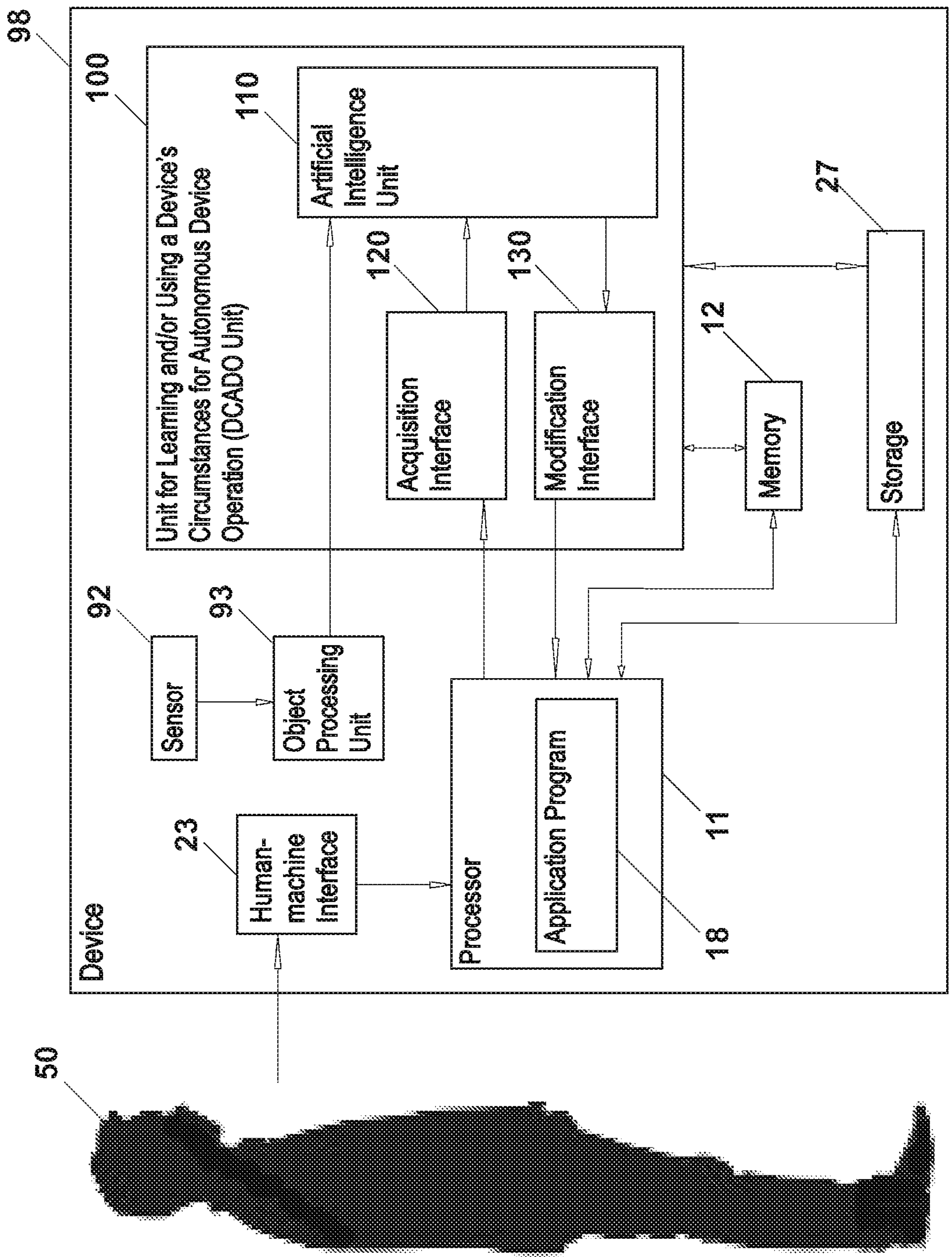

98
100
110
Unit for Learning and/or Using a Device's Circumstances for Autonomous Device Operation (DCADO Unit)
Artificial Intelligence Unit
120
130
Acquisition Interface
Modification Interface
27
12
Memory
Storage
92
93
Sensor
Object Processing Unit
23
Human-machine Interface
Device
Processor
Application Program
11
18
50

FIG. 2

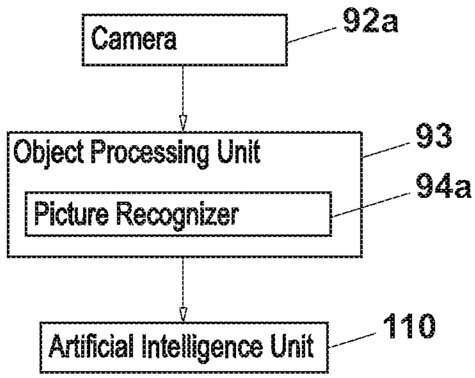


FIG. 3A

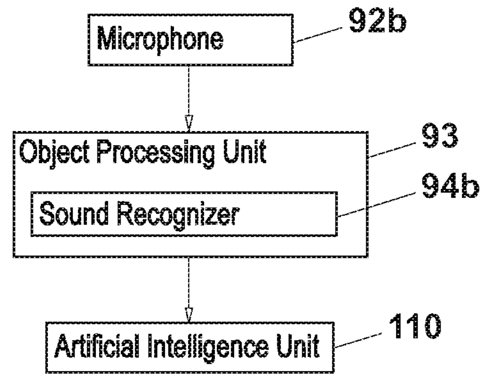


FIG. 3B

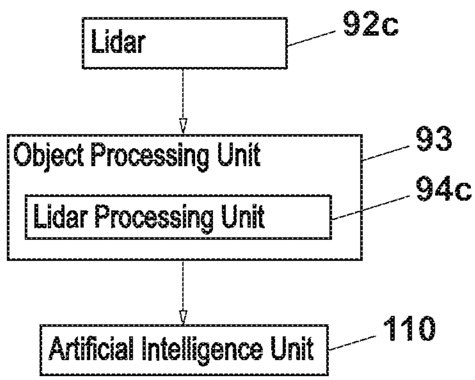


FIG. 3C

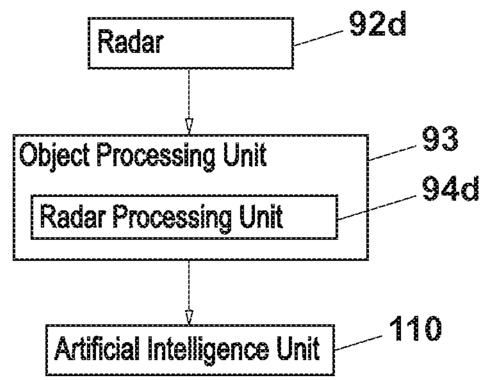


FIG. 3D

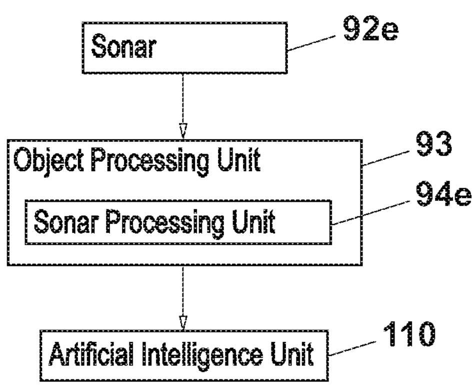


FIG. 3E

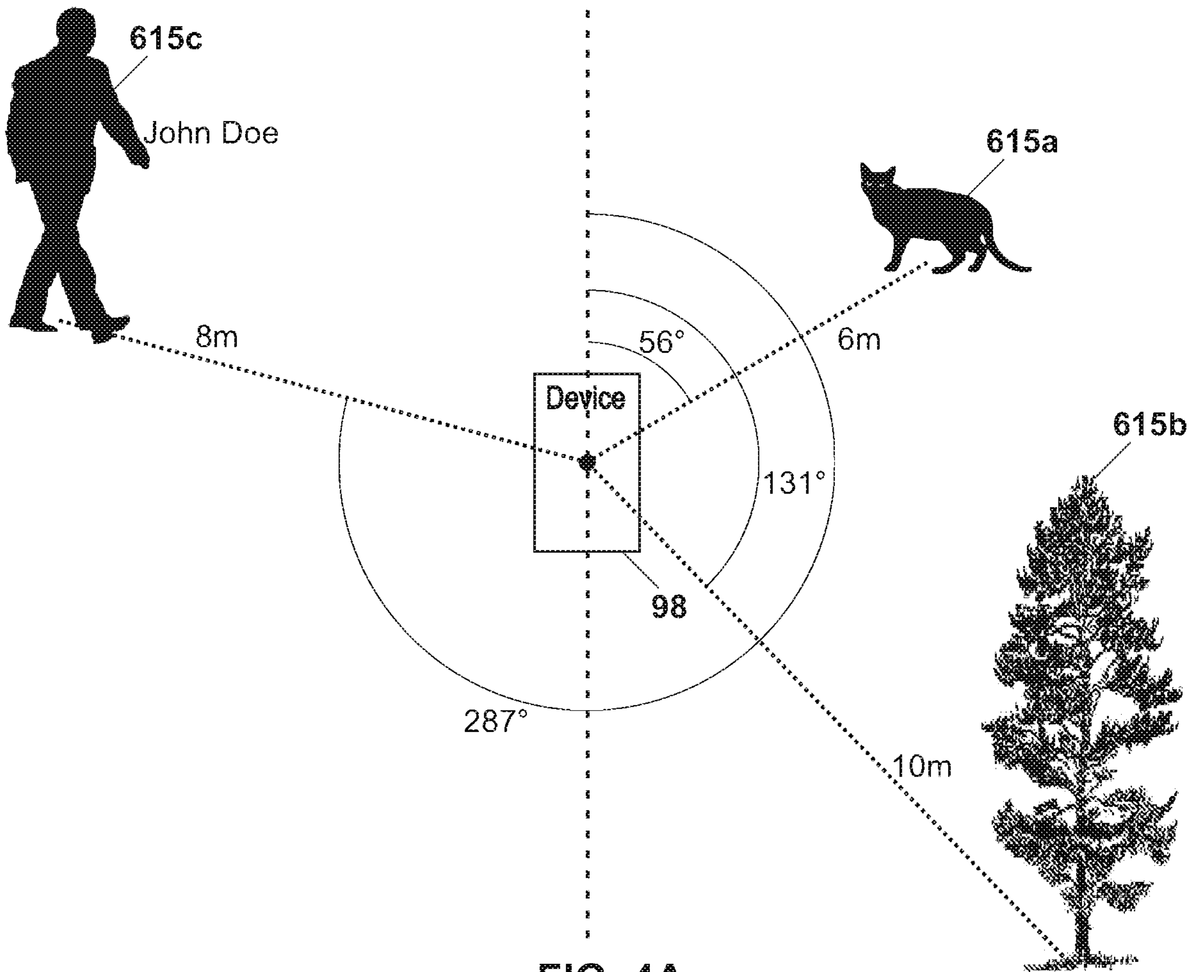

615c
John Doe
615a
8m
56°
6m
Device
131°
615b
98
287°
10m

FIG. 4A

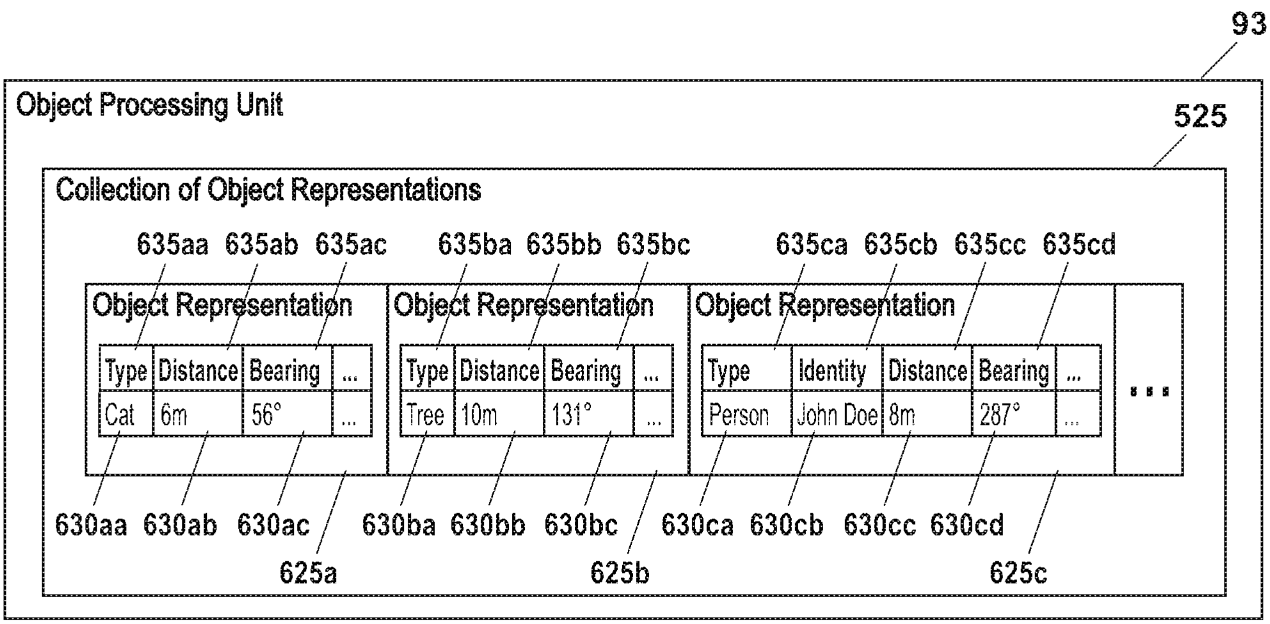

93
Object Processing Unit
525
Collection of Object Representations
635aa 635ab 635ac
635ba 635bb 635bc
635ca 635cb 635cc 635cd
Object Representation
Type Distance Bearing ...
Cat 6m 56° ...
Object Representation
Type Distance Bearing ...
Tree 10m 131° ...
Object Representation
Type Identity Distance Bearing ...
Person John Doe 8m 287° ...
...
630aa 630ab 630ac
630ba 630bb 630bc
630ca 630cb 630cc 630cd
625a
625b
625c

FIG. 4B

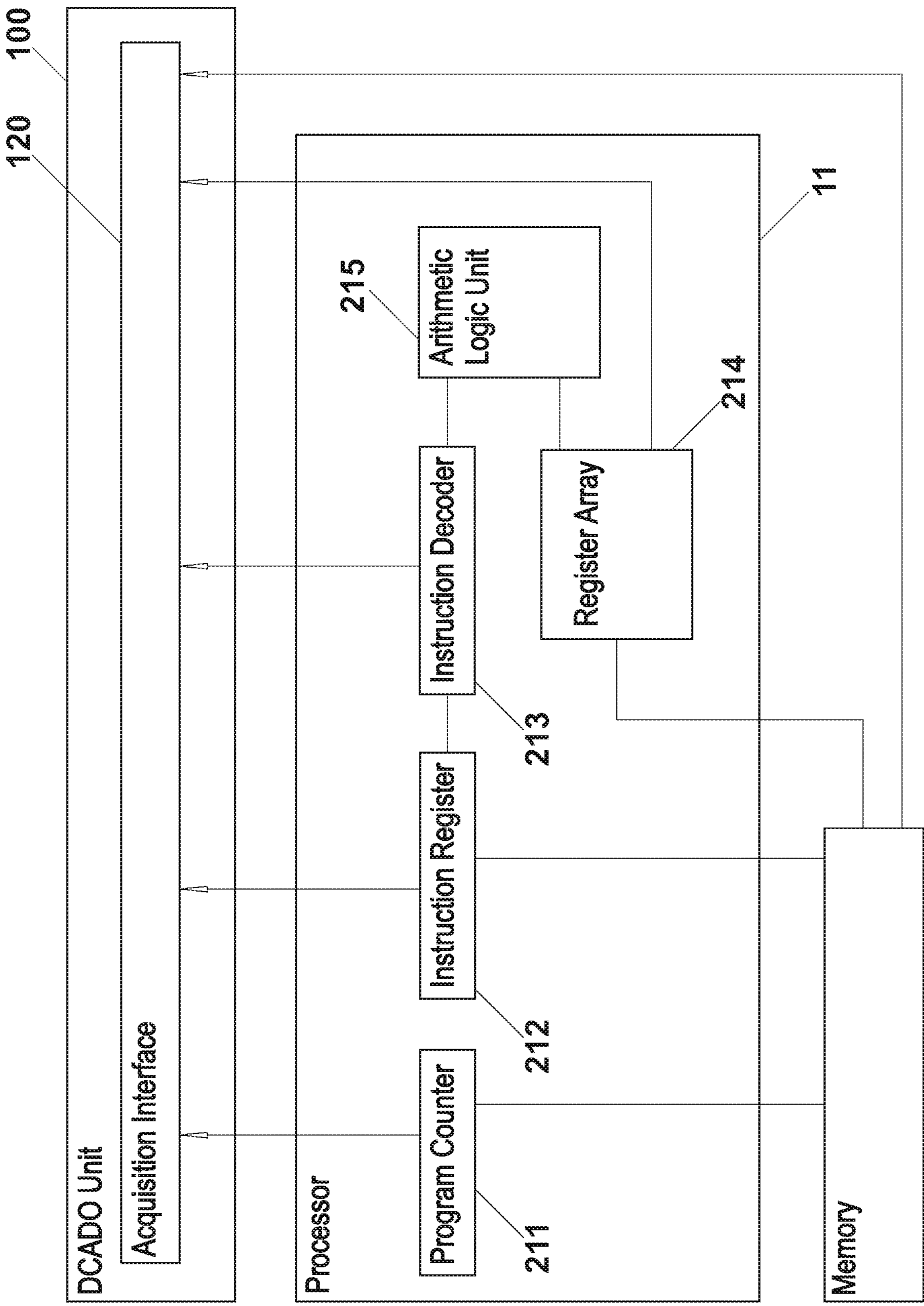

100
120
DCADO Unit
Acquisition Interface
Processor
Program Counter
Instruction Register
Instruction Decoder
Arithmetic Logic Unit
Register Array
Memory
211
212
213
214
215
11
12

FIG. 5

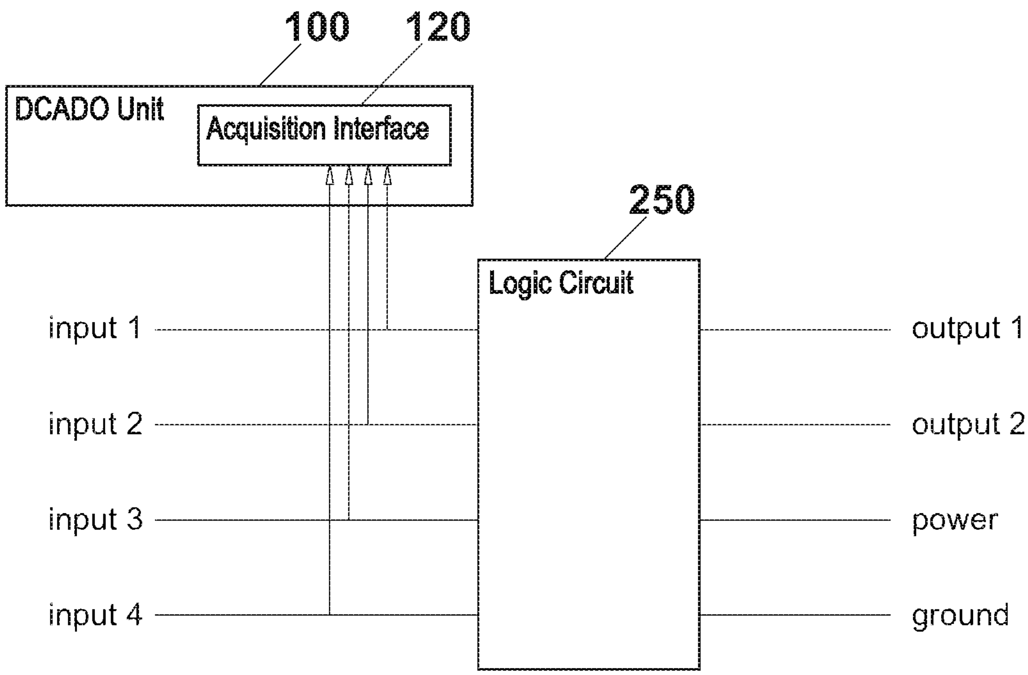

100
120
DCADO Unit
Acquisition Interface
250
Logic Circuit
input 1
input 2
input 3
input 4
output 1
output 2
power
ground

FIG. 6A

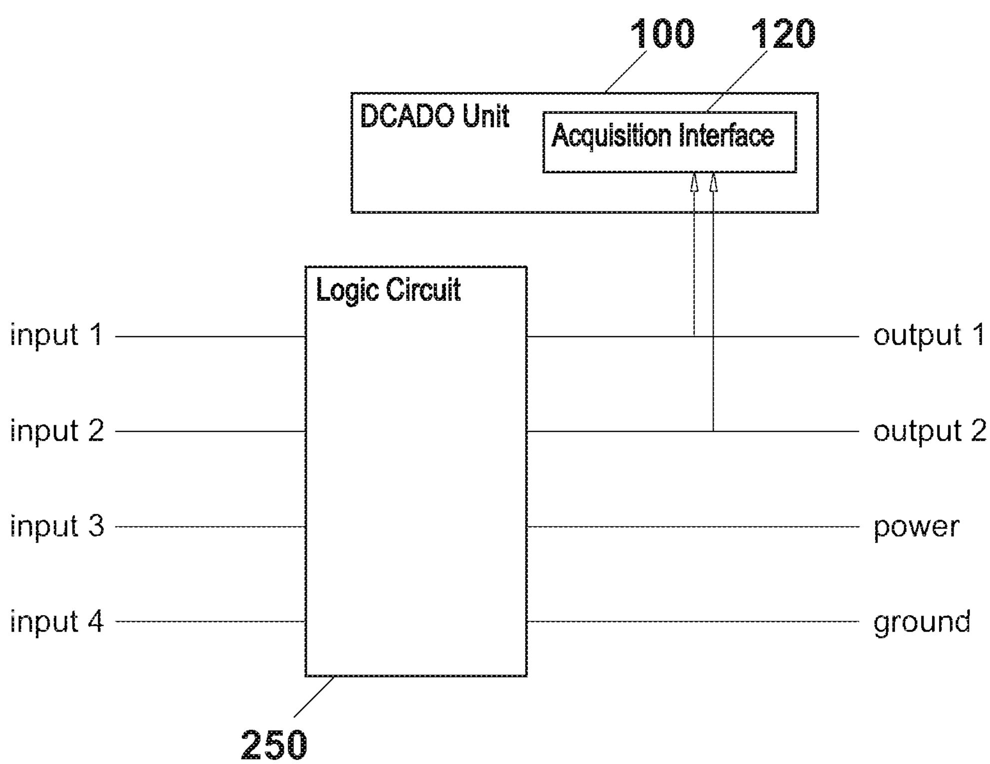

100
120
DCADO Unit
Acquisition Interface
Logic Circuit
input 1
input 2
input 3
input 4
output 1
output 2
power
ground
250

FIG. 6B

526

Instruction Set

Function (Parameter1, Parameter2, Parameter3, ...)

FIG. 7A

526

Instruction Set

SELECT Col1, Col2 ... FROM Tbl1, Tbl2 ... ;

FIG. 7B

526

Instruction Set

iconst_Value

FIG. 7C

526

Instruction Set

add Register1 Register2

FIG. 7D

526

Instruction Set

000000 00001 00010 00110 00000 100000

FIG. 7E

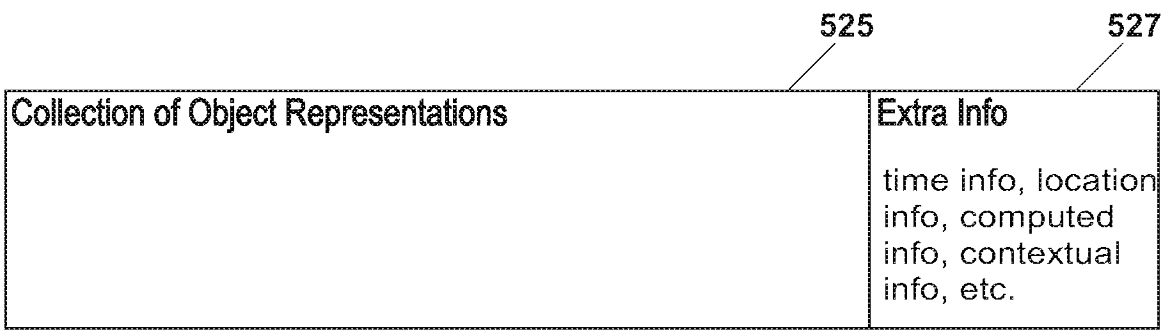

525
527
Collection of Object Representations
Extra Info
time info, location info, computed info, contextual info, etc.

FIG. 8A

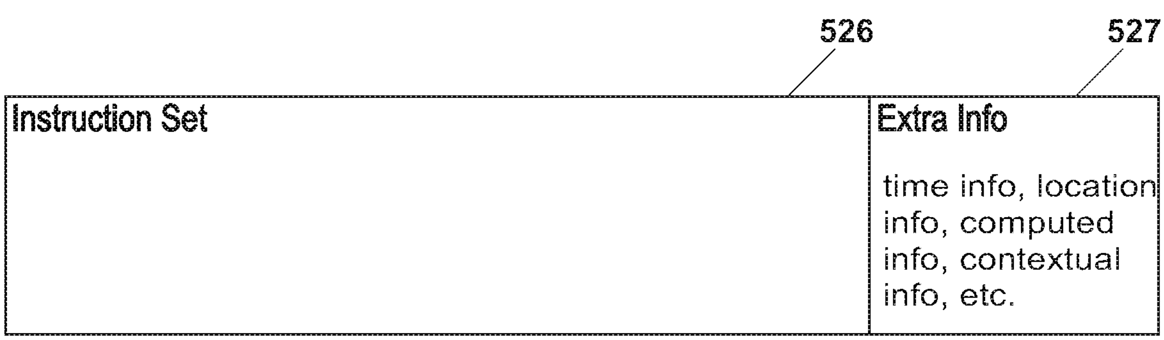

526
527
Instruction Set
Extra Info
time info, location info, computed info, contextual info, etc.

FIG. 8B

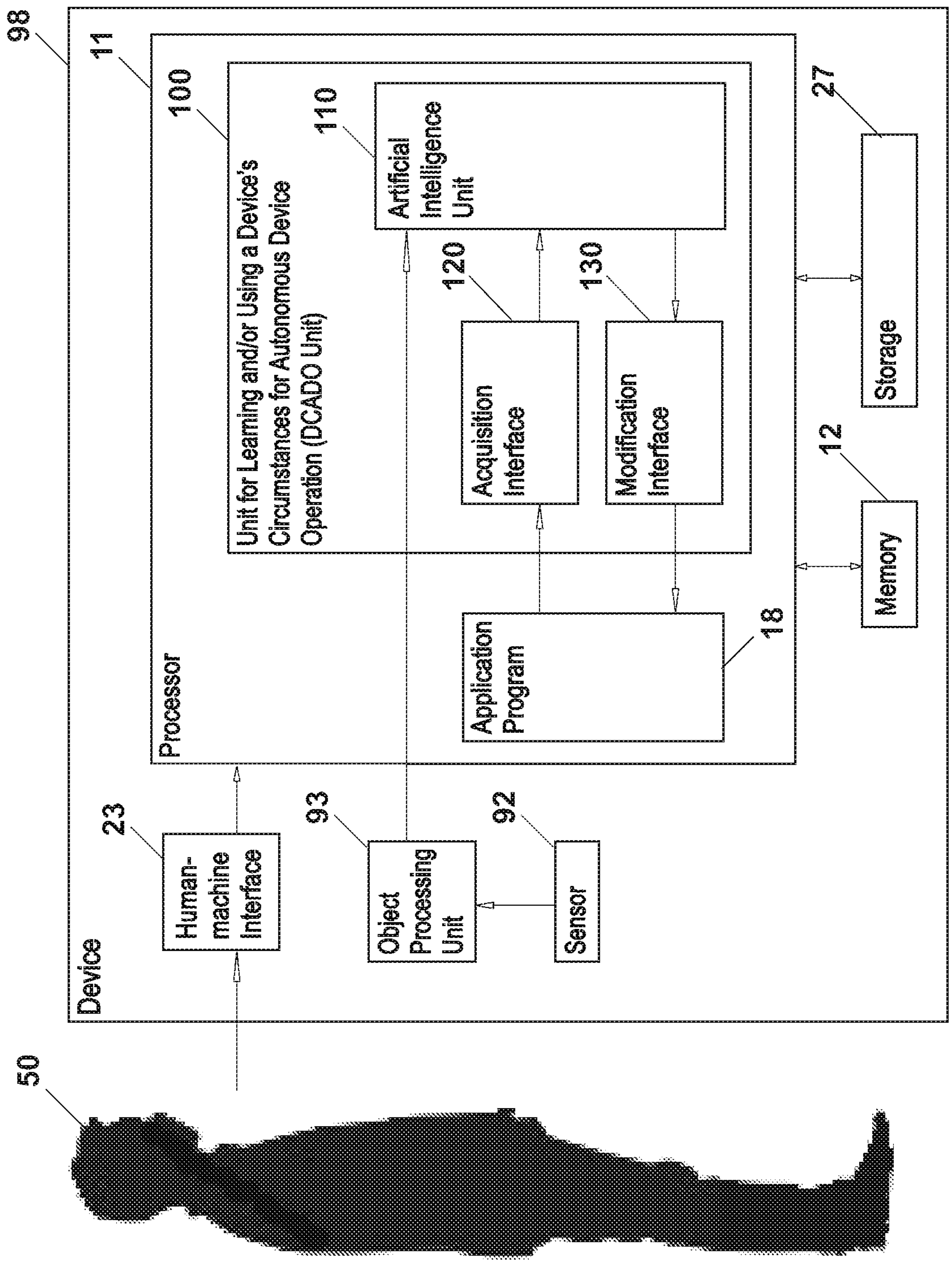

98
Device
11
Processor
100
Unit for Learning and/or Using a Device's Circumstances for Autonomous Device Operation (DCADO Unit)
110
Artificial Intelligence Unit
120
Acquisition Interface
130
Modification Interface
Application Program
18
27
Storage
12
Memory
23
Human-machine Interface
93
Object Processing Unit
92
Sensor
50

FIG. 9

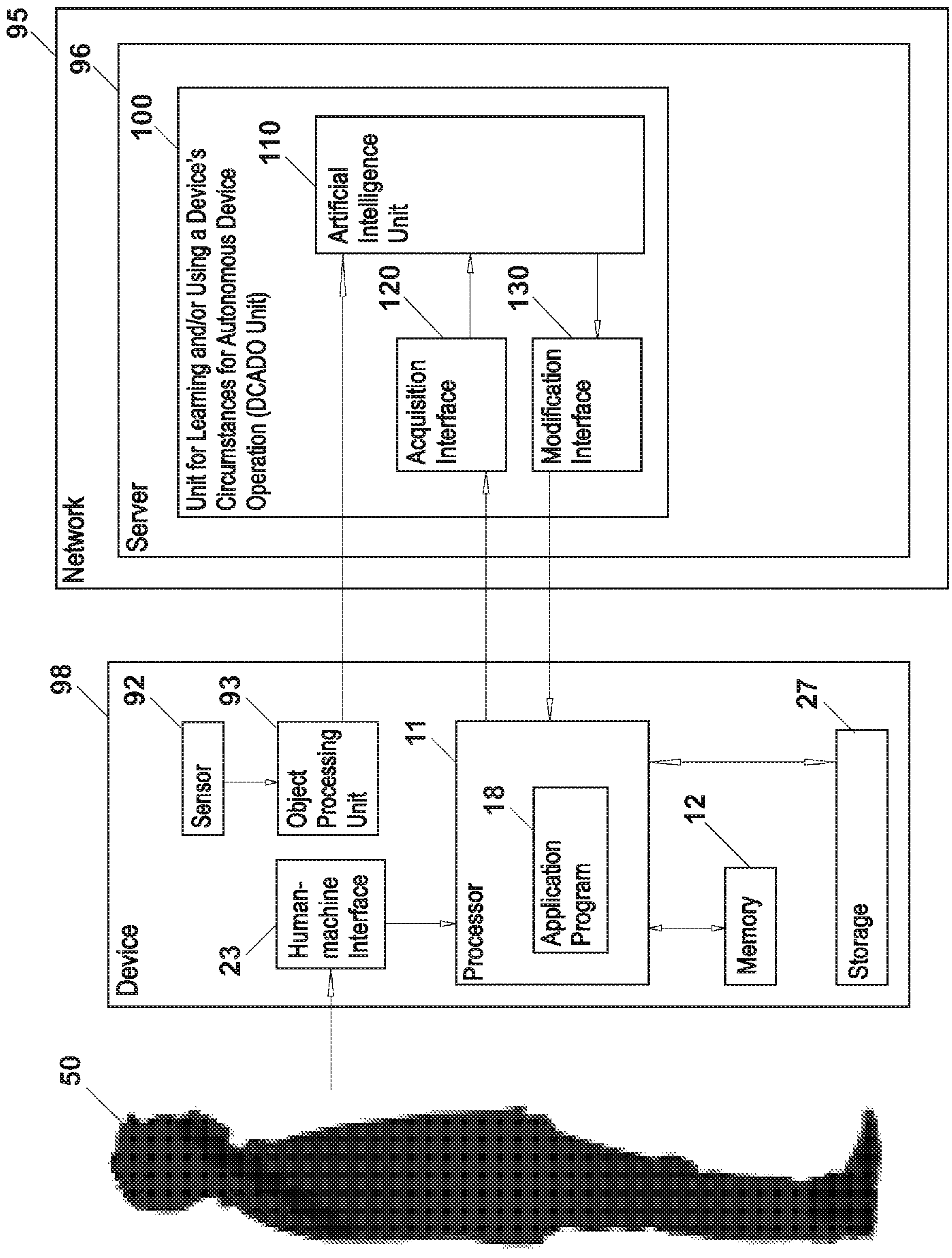
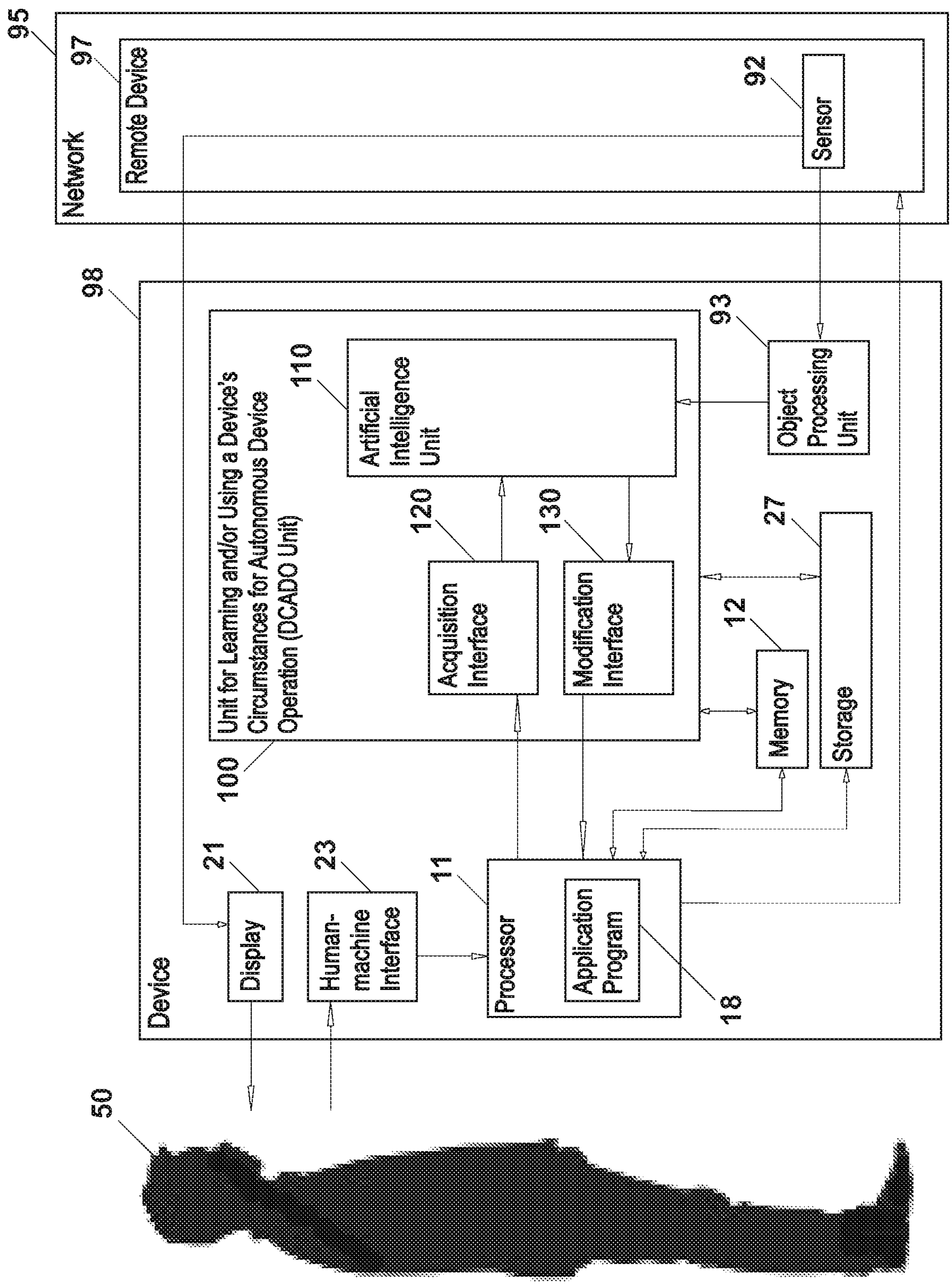

95
Network
96
Server
100
Unit for Learning and/or Using a Device's Circumstances for Autonomous Device Operation (DCADO Unit)
110
Artificial Intelligence Unit
120
Acquisition Interface
130
Modification Interface
98
Device
92
Sensor
93
Object Processing Unit
23
Human-machine Interface
11
Processor
18
Application Program
12
Memory
27
Storage
50

FIG. 10



95
Network
97
Remote Device
92
Sensor
98
Device
100
Unit for Learning and/or Using a Device's Circumstances for Autonomous Device Operation (DCADO Unit)
110
Artificial Intelligence Unit
120
Acquisition Interface
130
Modification Interface
93
Object Processing Unit
27
Storage
12
Memory
21
Display
23
Human-machine Interface
11
Processor
Application Program
18
50

FIG. 11

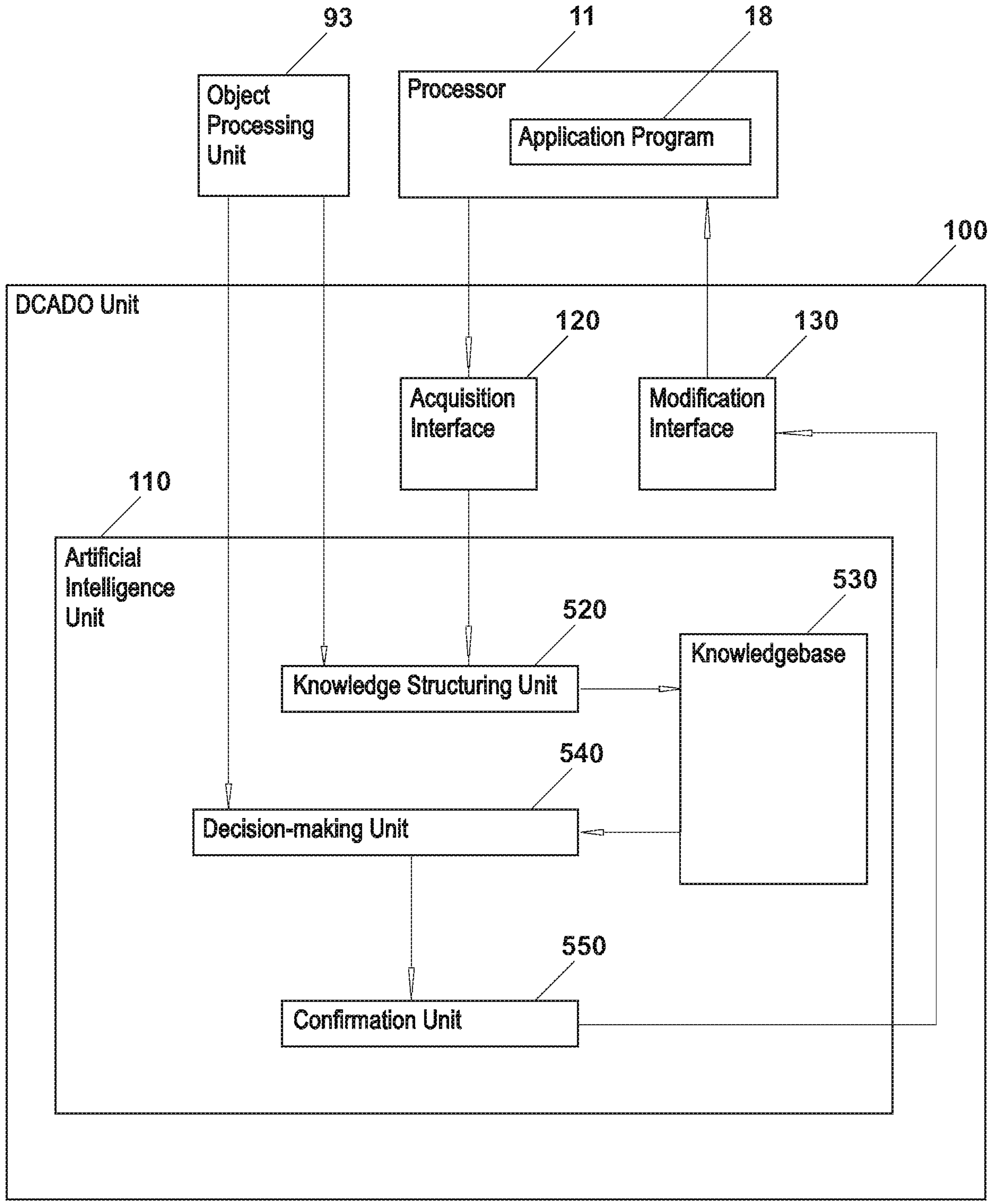
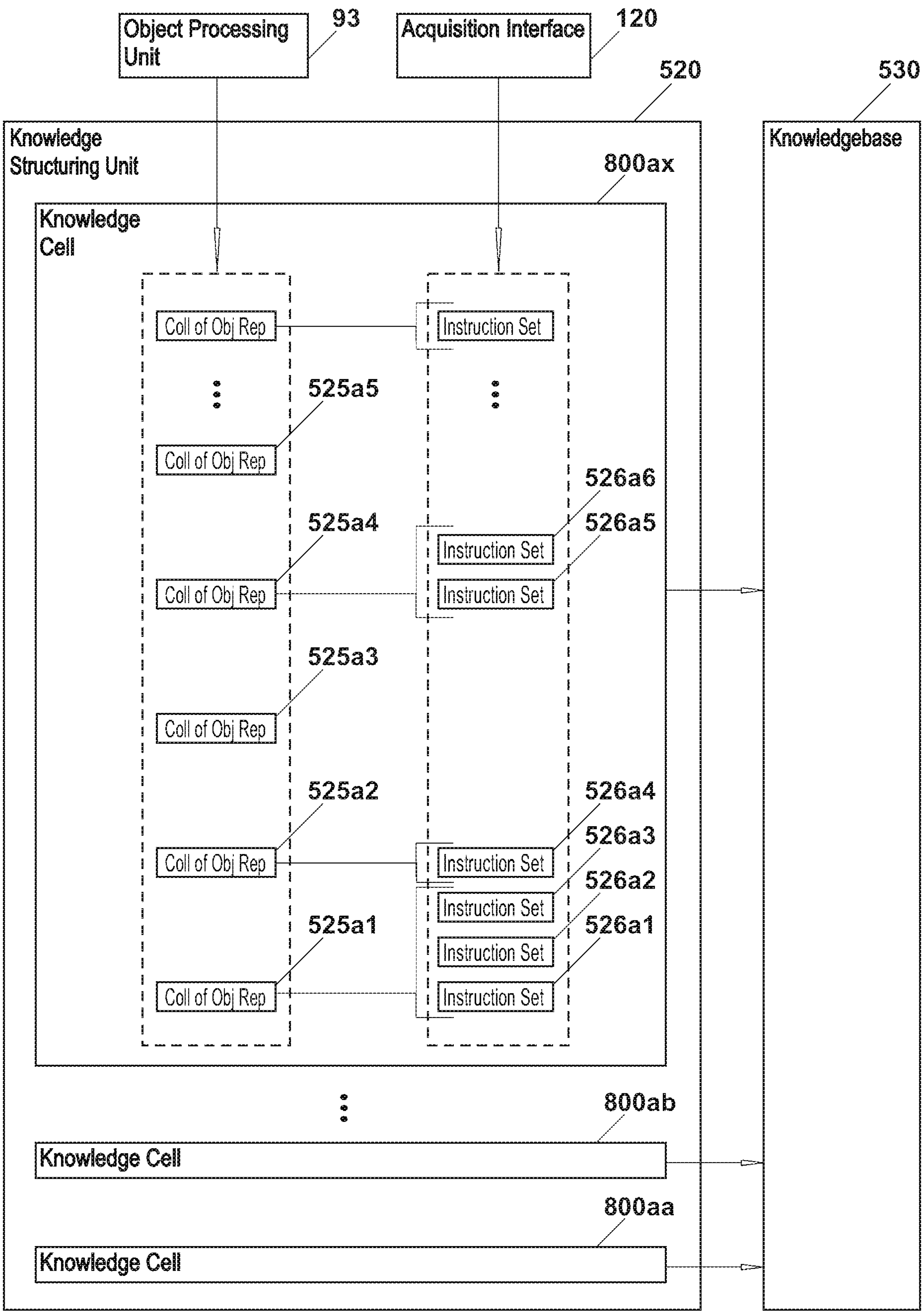

93
11
18
Object Processing Unit
Processor
Application Program
100
DCADO Unit
120
130
Acquisition Interface
Modification Interface
110
Artificial Intelligence Unit
530
520
Knowledgebase
Knowledge Structuring Unit
540
Decision-making Unit
550
Confirmation Unit

FIG. 12



Object Processing Unit
93
Acquisition Interface
120
520
530
Knowledge Structuring Unit
Knowledgebase
800ax
Knowledge Cell
Coll of Obj Rep
Instruction Set
525a5
Coll of Obj Rep
526a6
525a4
526a5
Instruction Set
Coll of Obj Rep
Instruction Set
525a3
Coll of Obj Rep
525a2
526a4
526a3
Coll of Obj Rep
Instruction Set
526a2
Instruction Set
525a1
526a1
Instruction Set
Coll of Obj Rep
Instruction Set
800ab
Knowledge Cell
800aa
Knowledge Cell

FIG. 13

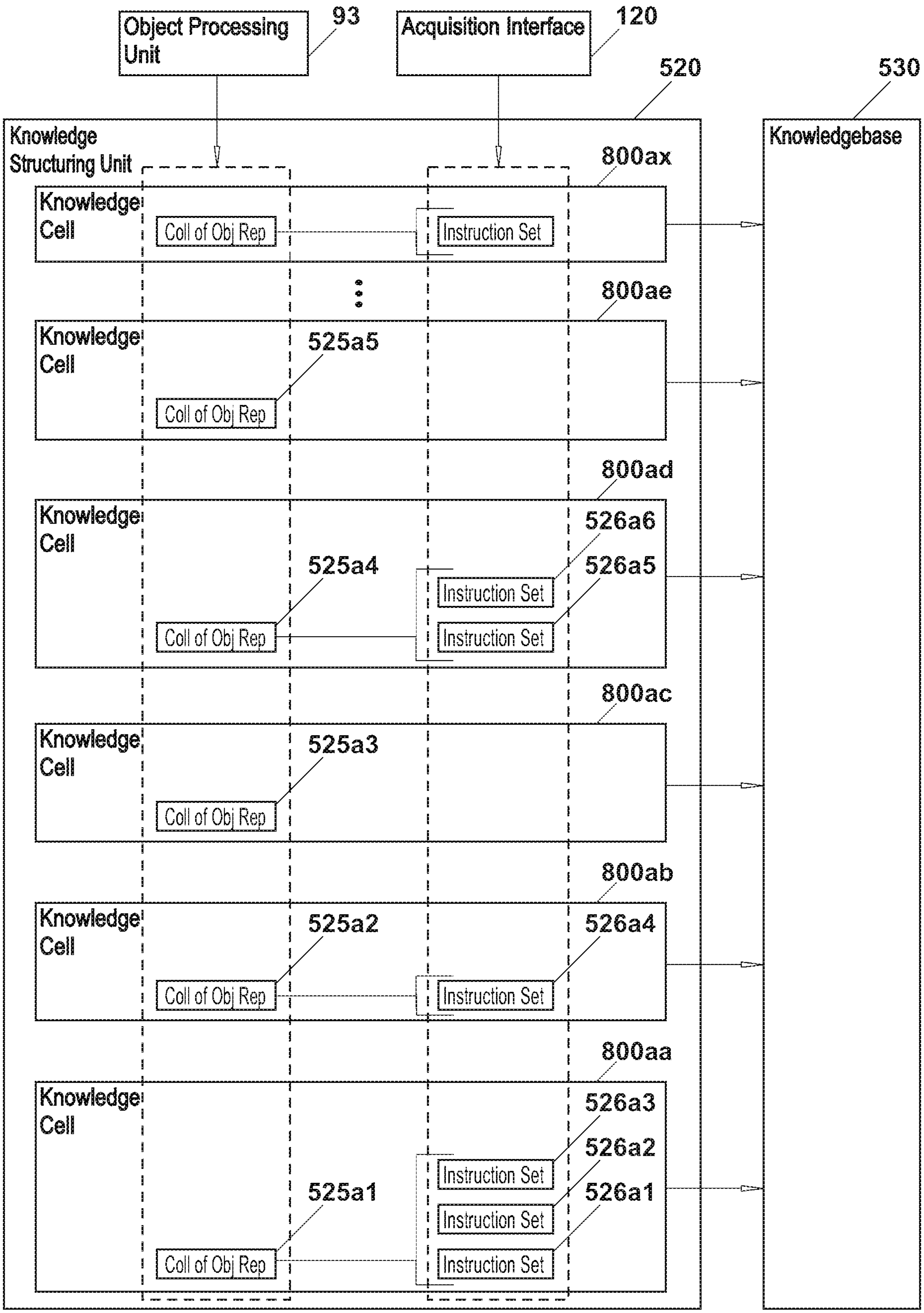
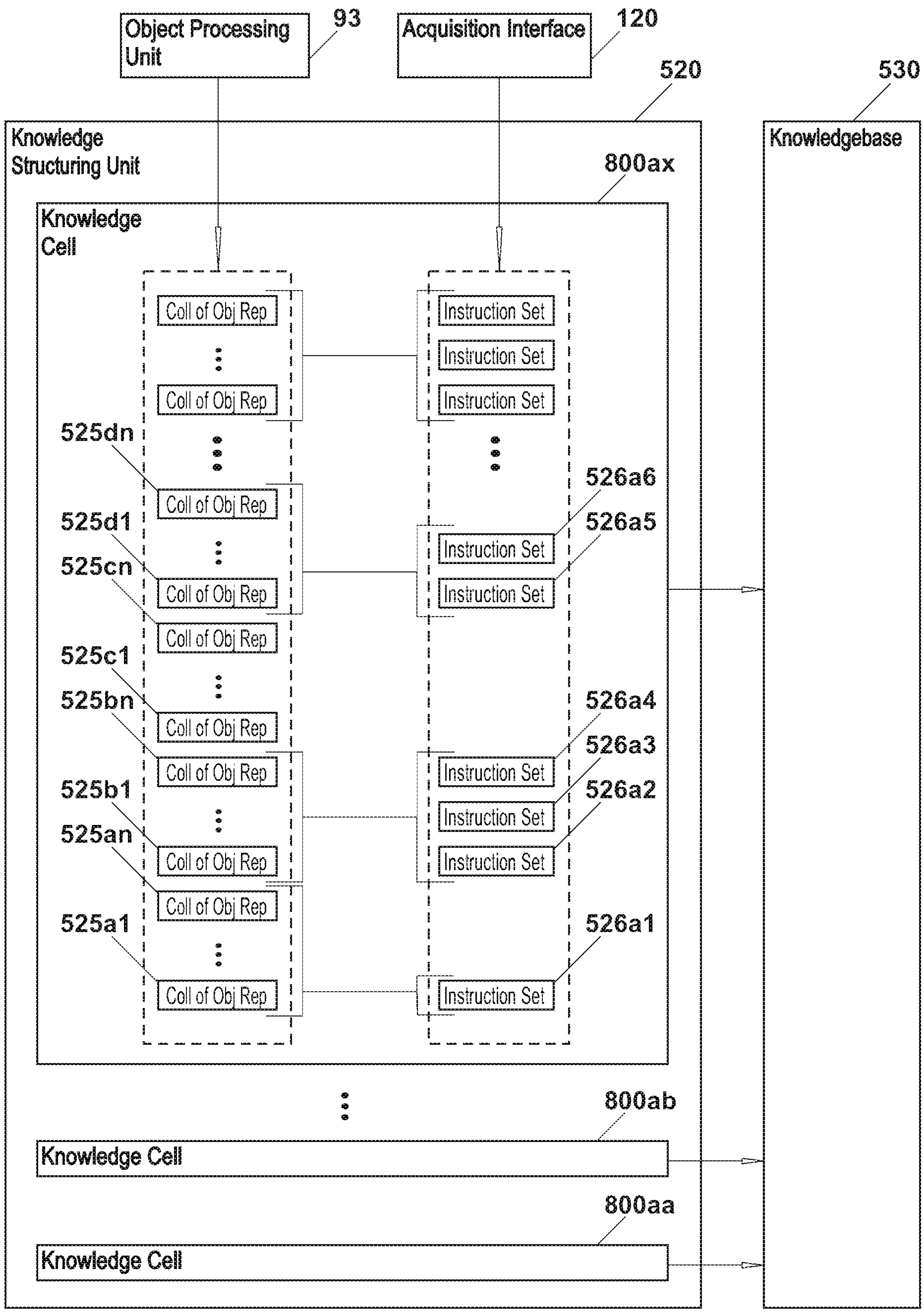
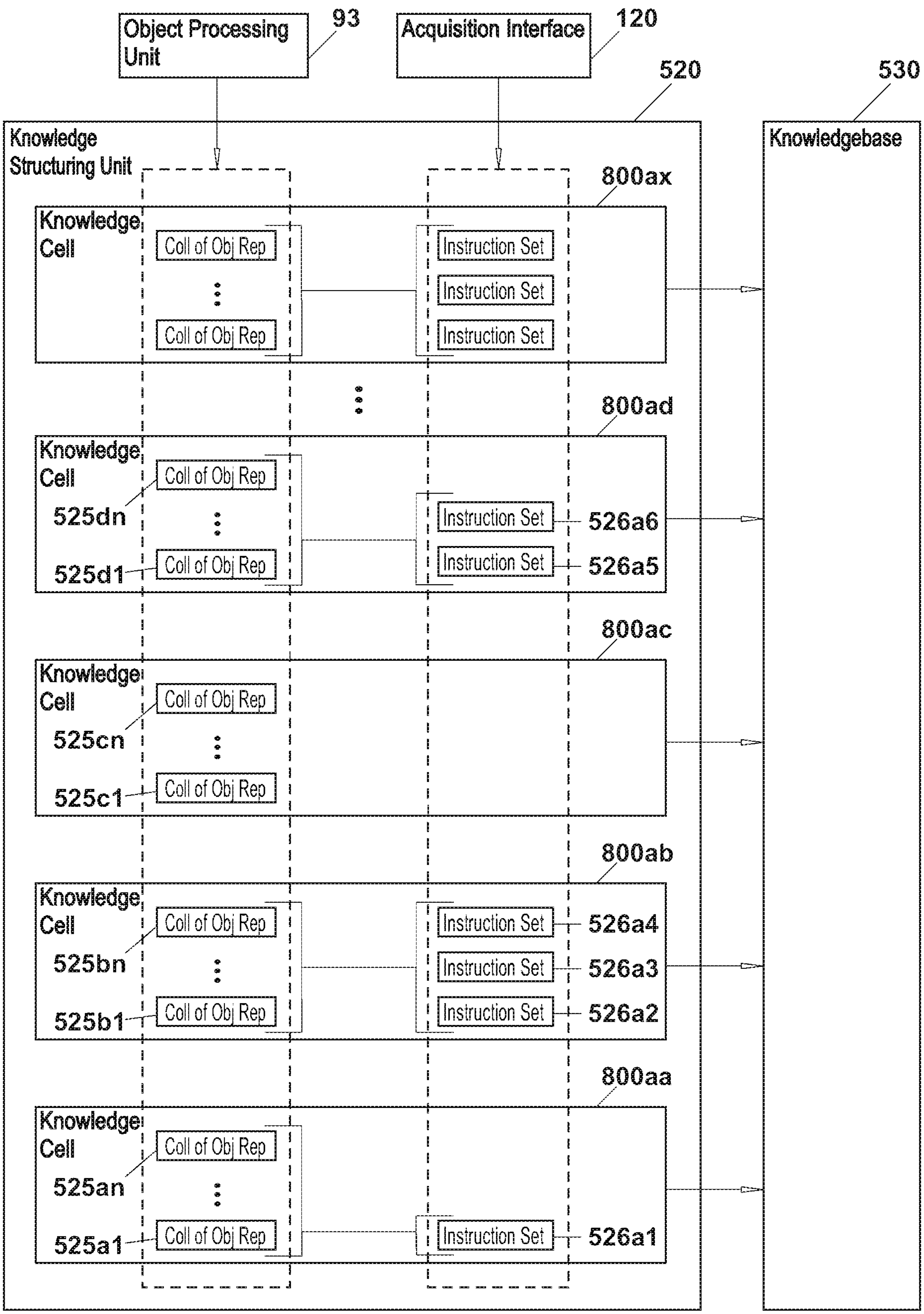

Object Processing Unit
93
Acquisition Interface
120
520
530
Knowledge Structuring Unit
Knowledgebase
800ax
Knowledge Cell
Coll of Obj Rep
Instruction Set
800ae
Knowledge Cell
525a5
Coll of Obj Rep
800ad
Knowledge Cell
526a6
525a4
526a5
Instruction Set
Coll of Obj Rep
Instruction Set
800ac
Knowledge Cell
525a3
Coll of Obj Rep
800ab
Knowledge Cell
525a2
526a4
Coll of Obj Rep
Instruction Set
800aa
Knowledge Cell
526a3
526a2
525a1
526a1
Instruction Set
Instruction Set
Coll of Obj Rep
Instruction Set

FIG. 14



Object Processing Unit
93
Acquisition Interface
120
520
530
Knowledge Structuring Unit
800ax
Knowledgebase
Knowledge Cell
Coll of Obj Rep
Instruction Set
525dn
525d1
525cn
525c1
525bn
525b1
525an
525a1
526a6
526a5
526a4
526a3
526a2
526a1
800ab
Knowledge Cell
800aa
Knowledge Cell

FIG. 15



Object Processing Unit
93
Acquisition Interface
120
520
530
Knowledge Structuring Unit
Knowledgebase
800ax
Knowledge Cell
Coll of Obj Rep
Coll of Obj Rep
Instruction Set
Instruction Set
Instruction Set
800ad
Knowledge Cell
525dn
525d1
Coll of Obj Rep
Coll of Obj Rep
Instruction Set
Instruction Set
526a6
526a5
800ac
Knowledge Cell
525cn
525c1
Coll of Obj Rep
Coll of Obj Rep
800ab
Knowledge Cell
525bn
525b1
Coll of Obj Rep
Coll of Obj Rep
Instruction Set
Instruction Set
Instruction Set
526a4
526a3
526a2
800aa
Knowledge Cell
525an
525a1
Coll of Obj Rep
Coll of Obj Rep
Instruction Set
526a1

FIG. 16

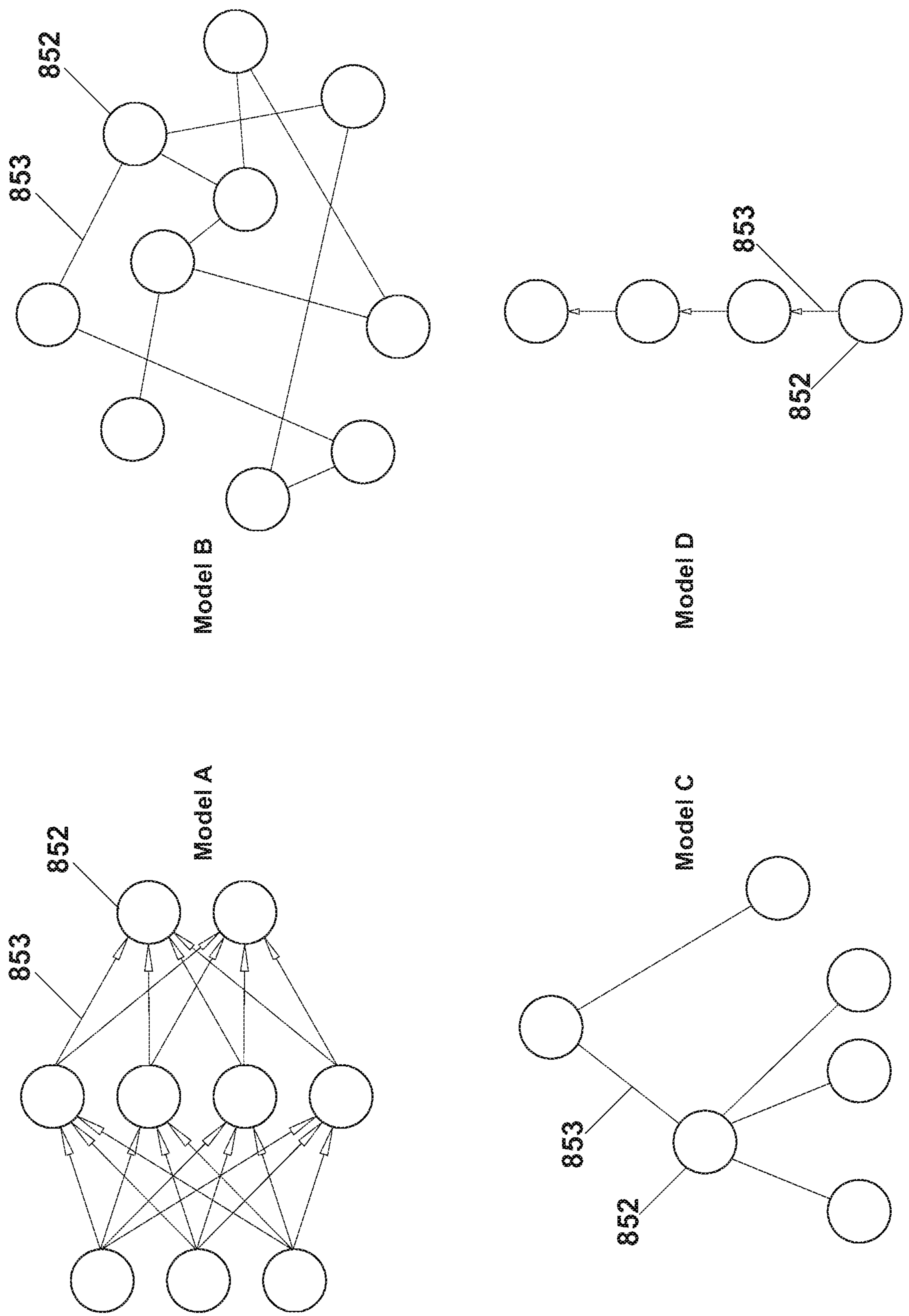


FIG. 17

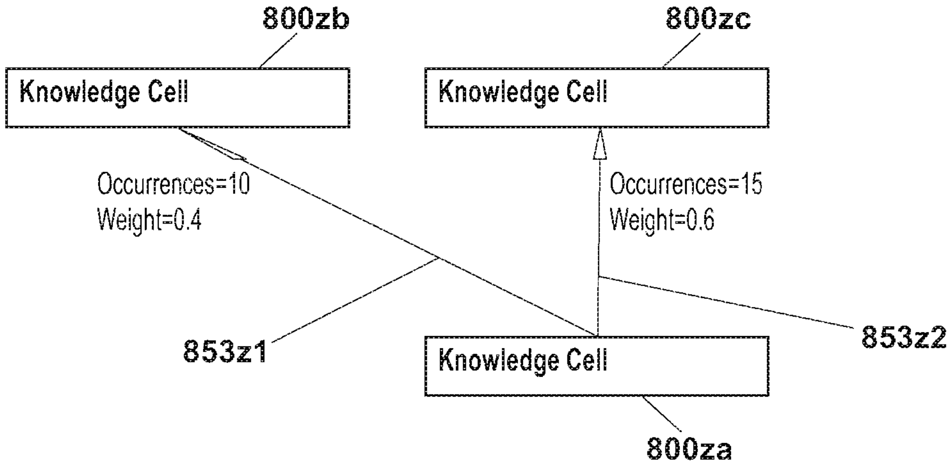


FIG. 18A

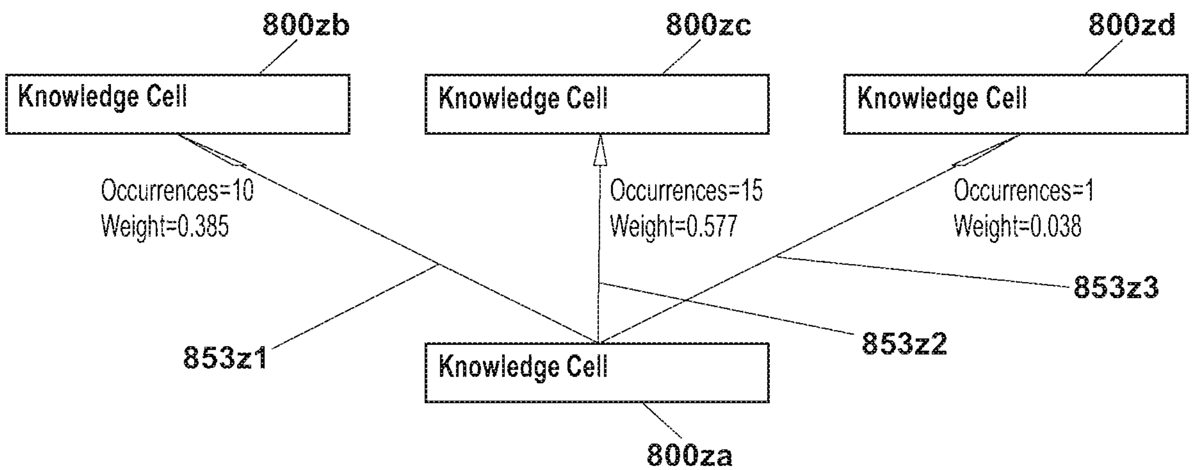


FIG. 18B

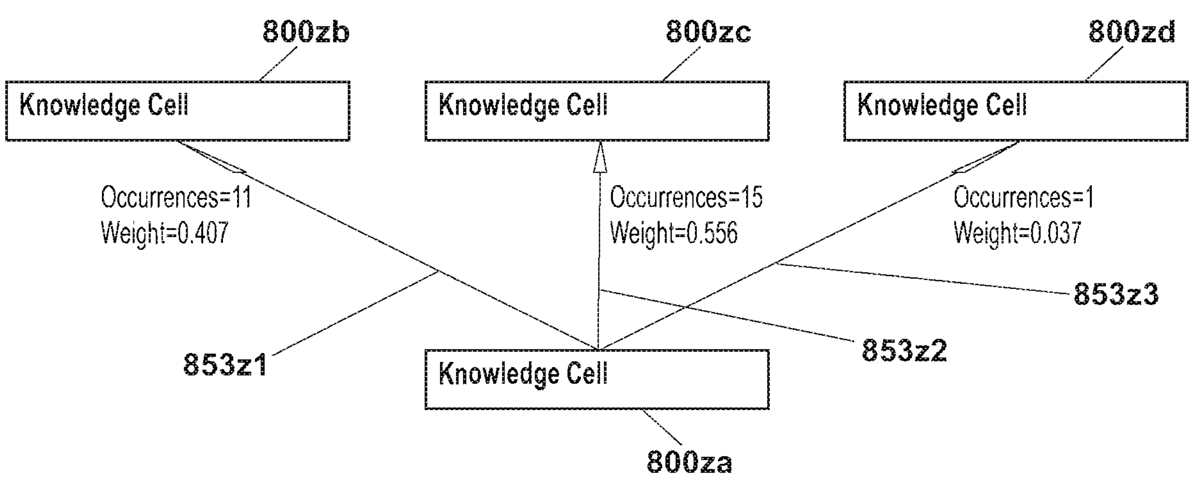


FIG. 18C

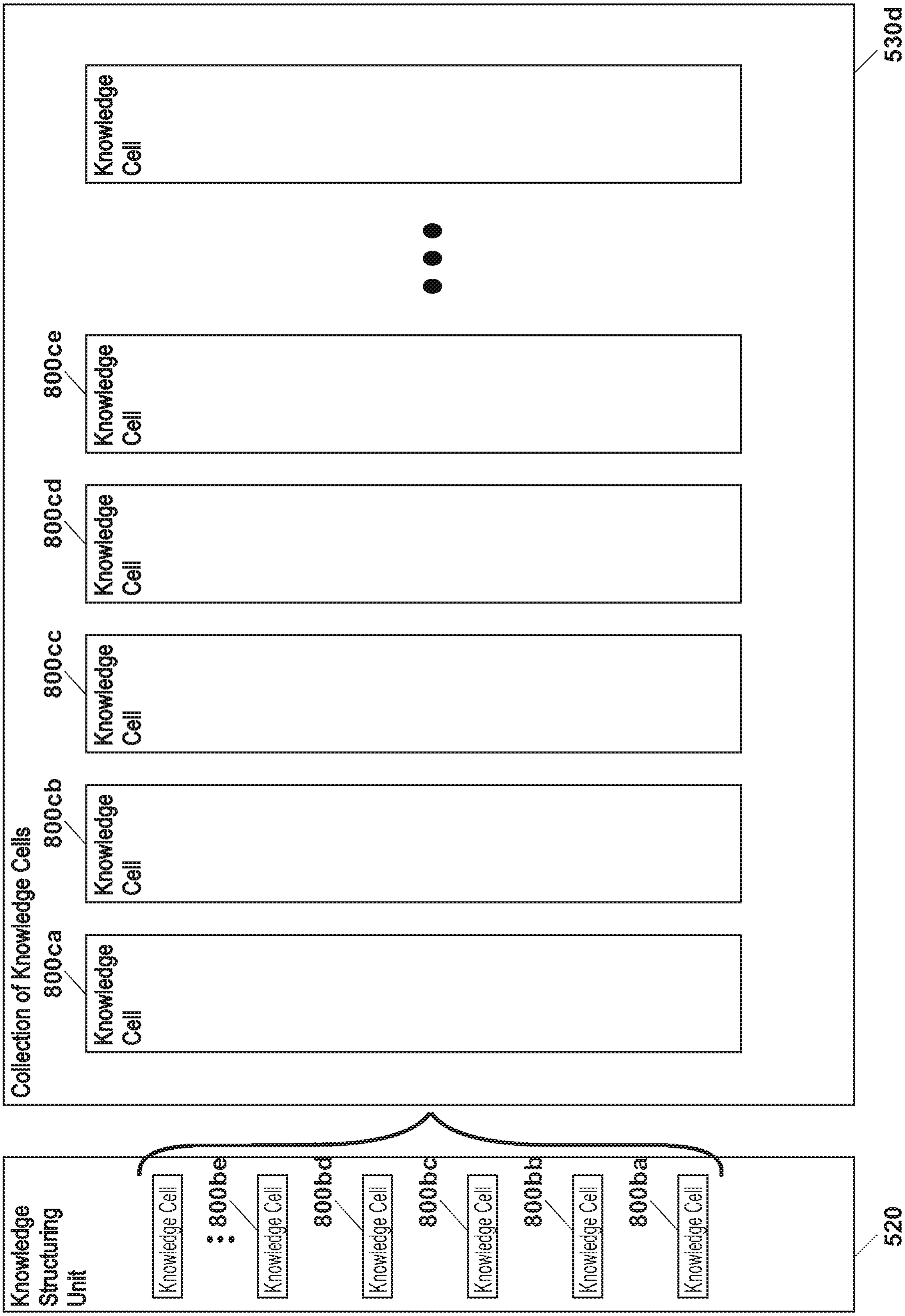

Collection of Knowledge Cells
800ca
Knowledge Cell
800cb
Knowledge Cell
800cc
Knowledge Cell
800cd
Knowledge Cell
800ce
Knowledge Cell
Knowledge Cell
530d
Knowledge Structuring Unit
Knowledge Cell
800be
Knowledge Cell
800bd
Knowledge Cell
800bc
Knowledge Cell
800bb
Knowledge Cell
800ba
Knowledge Cell
520

FIG. 19

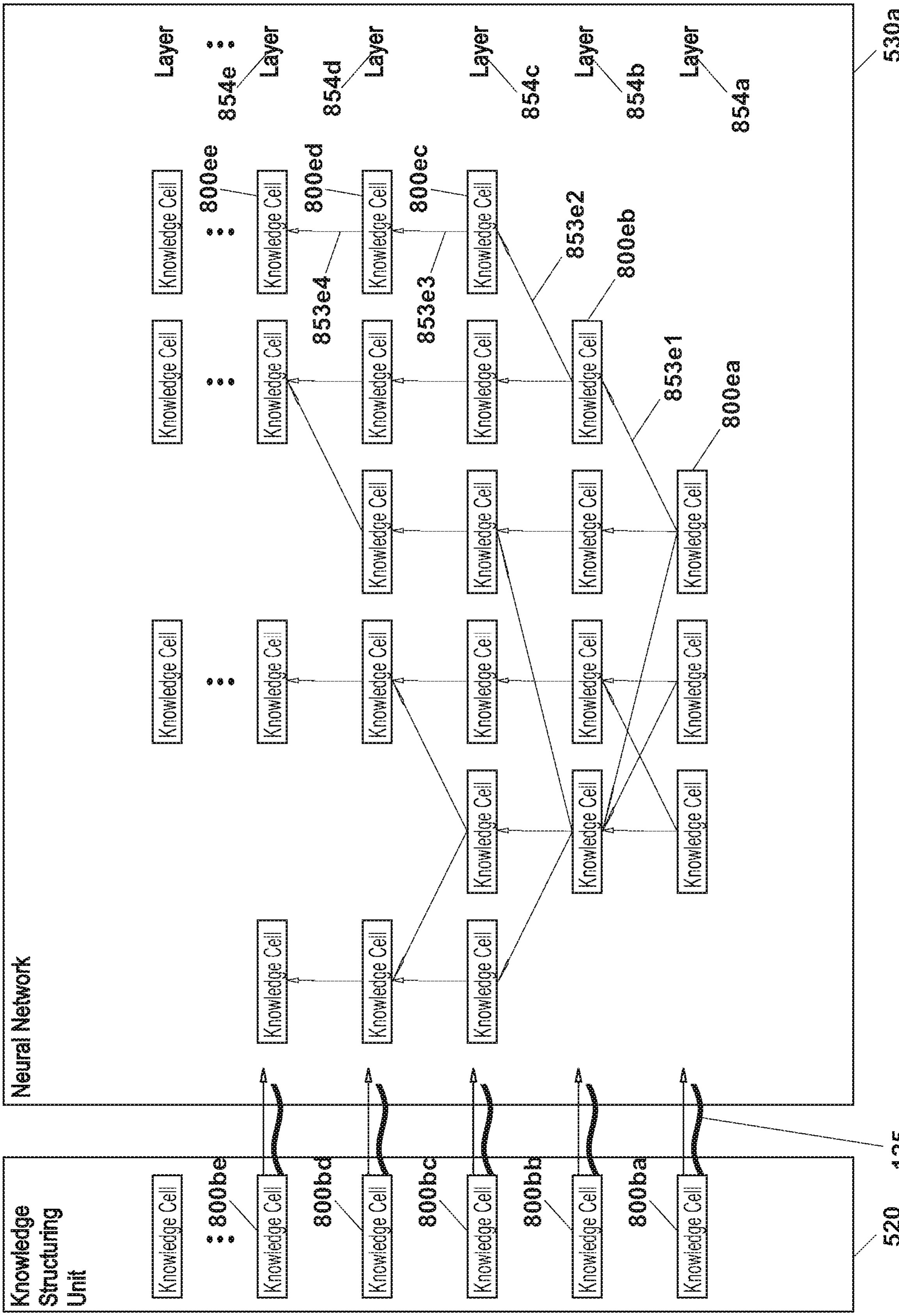

Knowledge Structuring Unit
Neural Network
Knowledge Cell
Layer
800ba
800bb
800bc
800bd
800be
800ea
800eb
800ec
800ed
800ee
853e1
853e2
853e3
853e4
854a
854b
854c
854d
854e
125
520
530a

FIG. 20

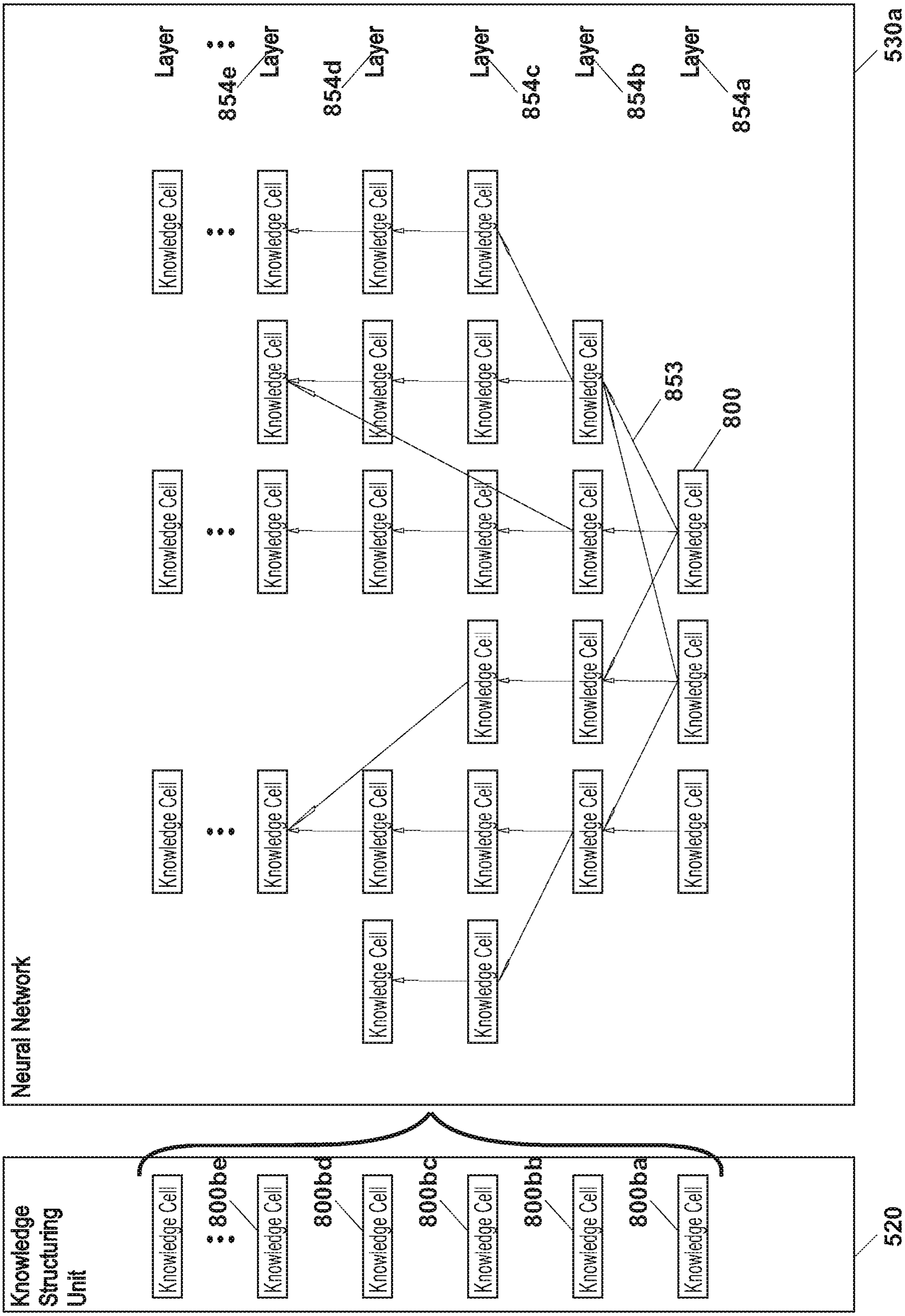

Neural Network
Layer
854e
Layer
854d
Layer
854c
Layer
854b
Layer
854a
Knowledge Cell
853
800
530a
Knowledge Structuring Unit
800be
800bd
800bc
800bb
800ba
520

FIG. 21

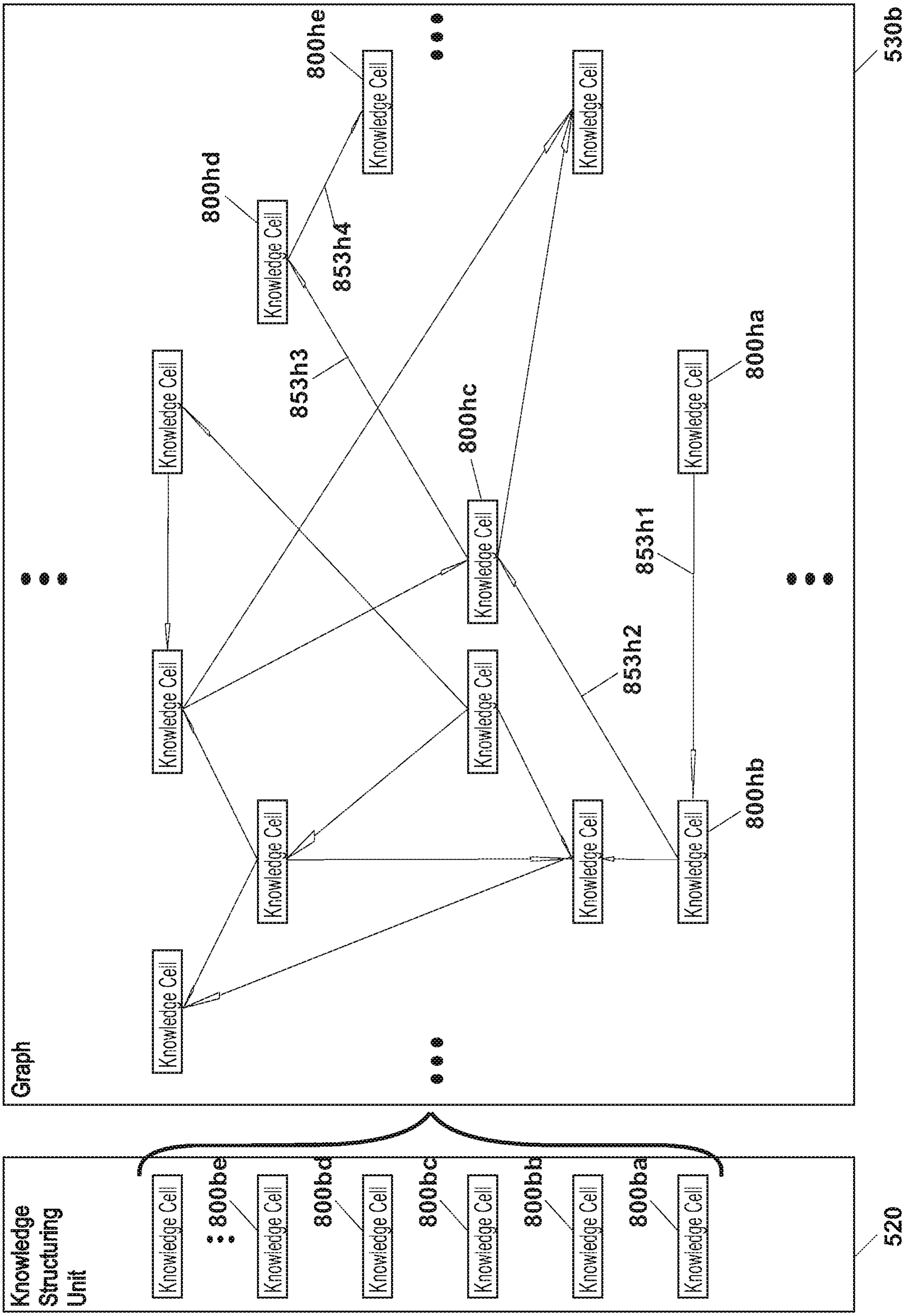
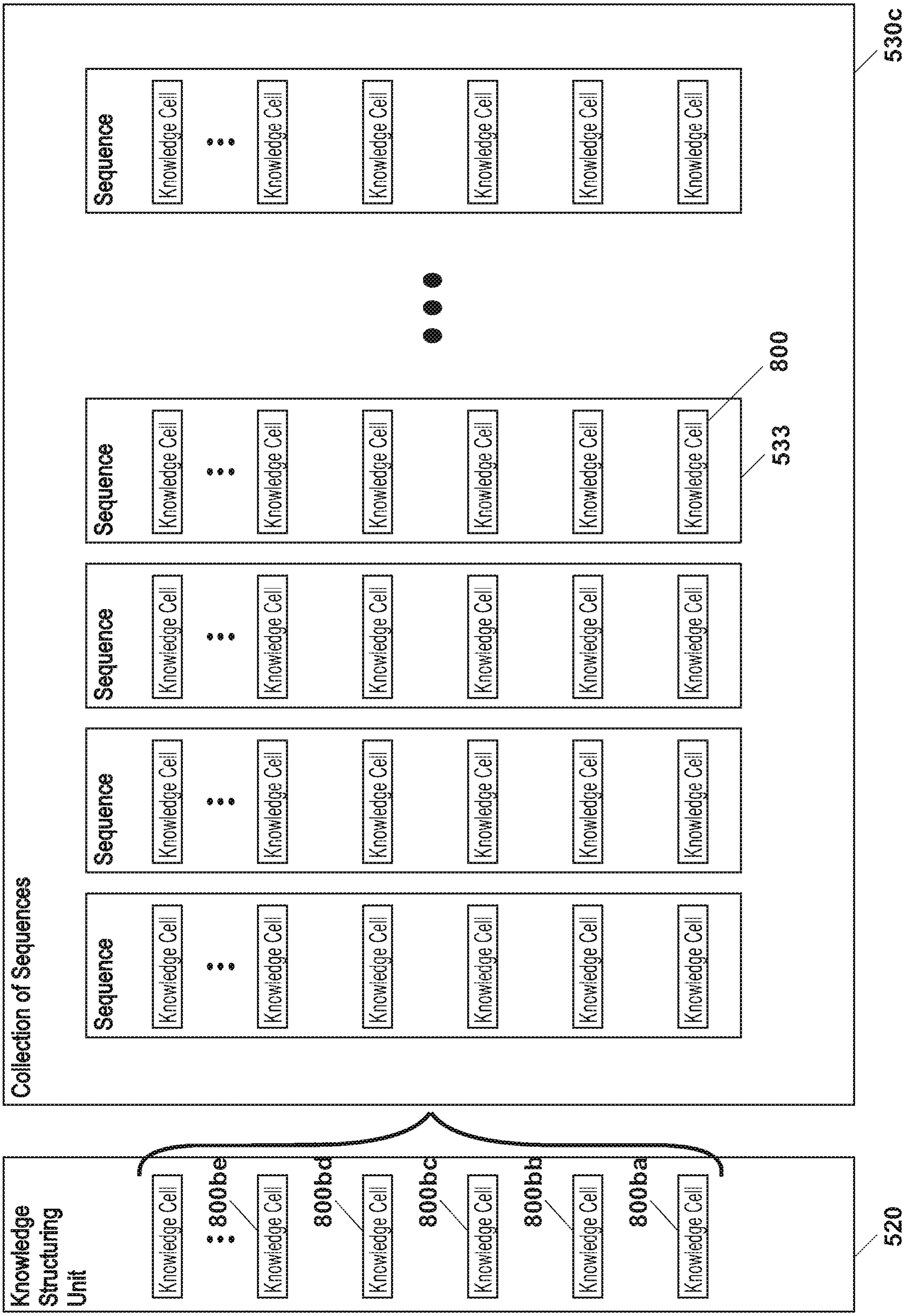

Graph
Knowledge Cell
800he
800hd
853h4
853h3
800hc
800ha
853h1
853h2
800hb
530b
Knowledge Structuring Unit
800be
800bd
800bc
800bb
800ba
520

FIG. 22



Collection of Sequences
Sequence
Knowledge Cell
800
533
530c
Knowledge Structuring Unit
800be
800bd
800bc
800bb
800ba
520

FIG. 23

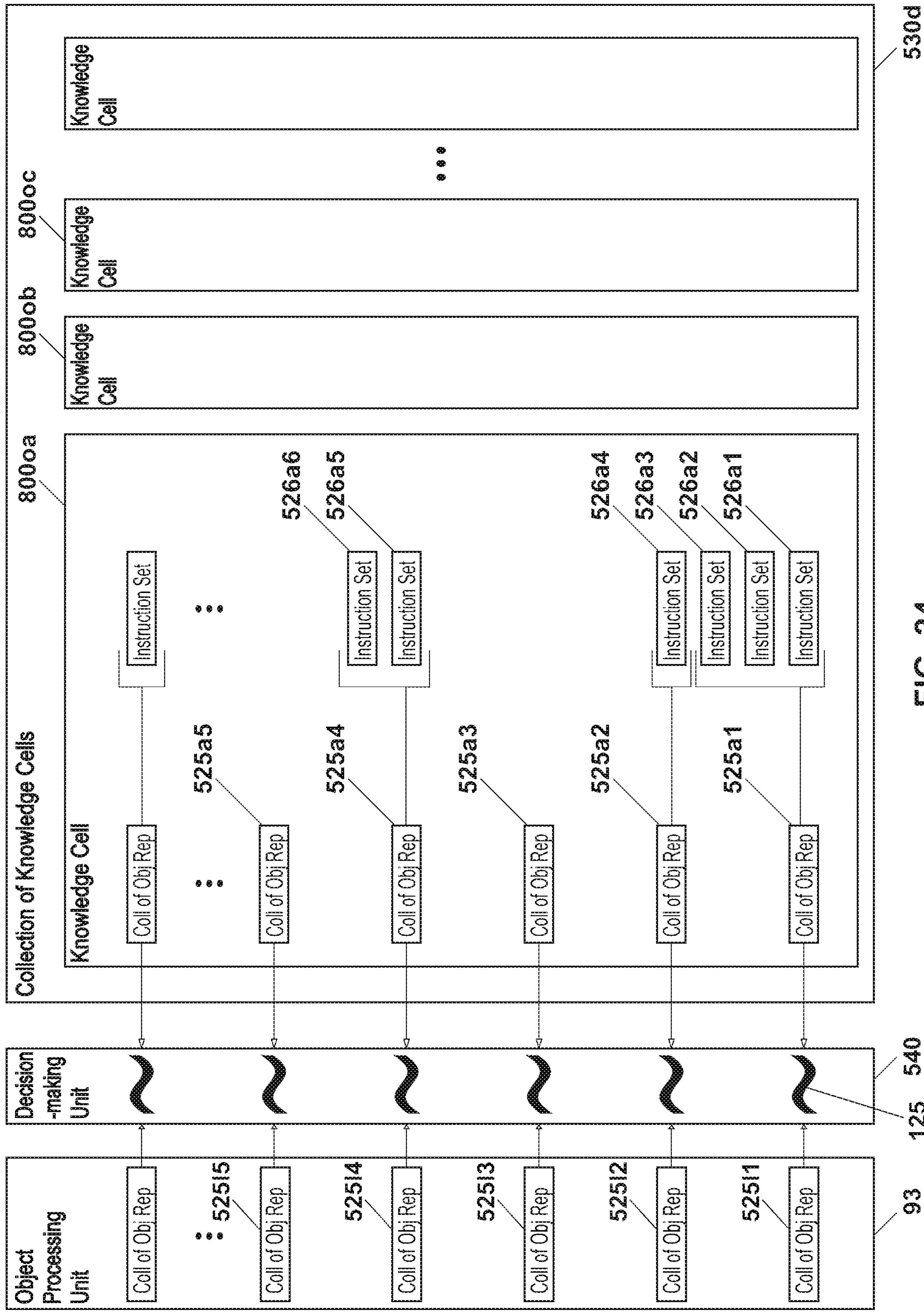

Object Processing Unit
Coll of Obj Rep
525l1
525l2
525l3
525l4
525l5
93
Decision -making Unit
125
540
Collection of Knowledge Cells
Knowledge Cell
Coll of Obj Rep
525a1
525a2
525a3
525a4
525a5
Instruction Set
526a1
526a2
526a3
526a4
526a5
526a6
800oa
800ob
800oc
Knowledge Cell
530d

FIG. 24

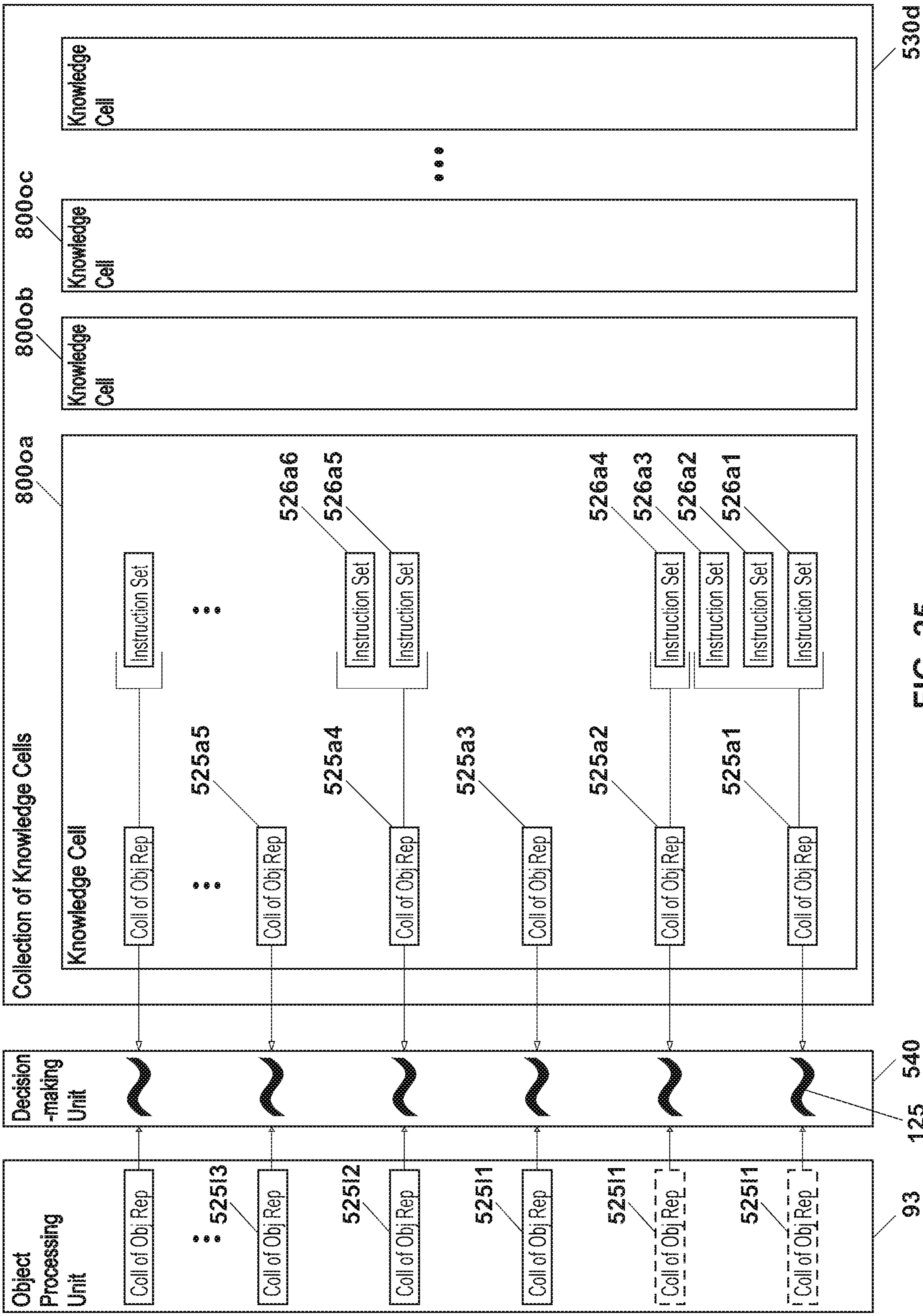

Object Processing Unit
Coll of Obj Rep
Coll of Obj Rep
Coll of Obj Rep
Coll of Obj Rep
Coll of Obj Rep
Coll of Obj Rep
525l3
525l2
525l1
525l1
525l1
93
Decision -making Unit
125
540
Collection of Knowledge Cells
Knowledge Cell
Coll of Obj Rep
Coll of Obj Rep
Coll of Obj Rep
Coll of Obj Rep
Coll of Obj Rep
Coll of Obj Rep
525a5
525a4
525a3
525a2
525a1
Instruction Set
Instruction Set
Instruction Set
Instruction Set
Instruction Set
Instruction Set
Instruction Set
526a6
526a5
526a4
526a3
526a2
526a1
800oa
Knowledge Cell
800ob
Knowledge Cell
800oc
Knowledge Cell
530d

FIG. 25

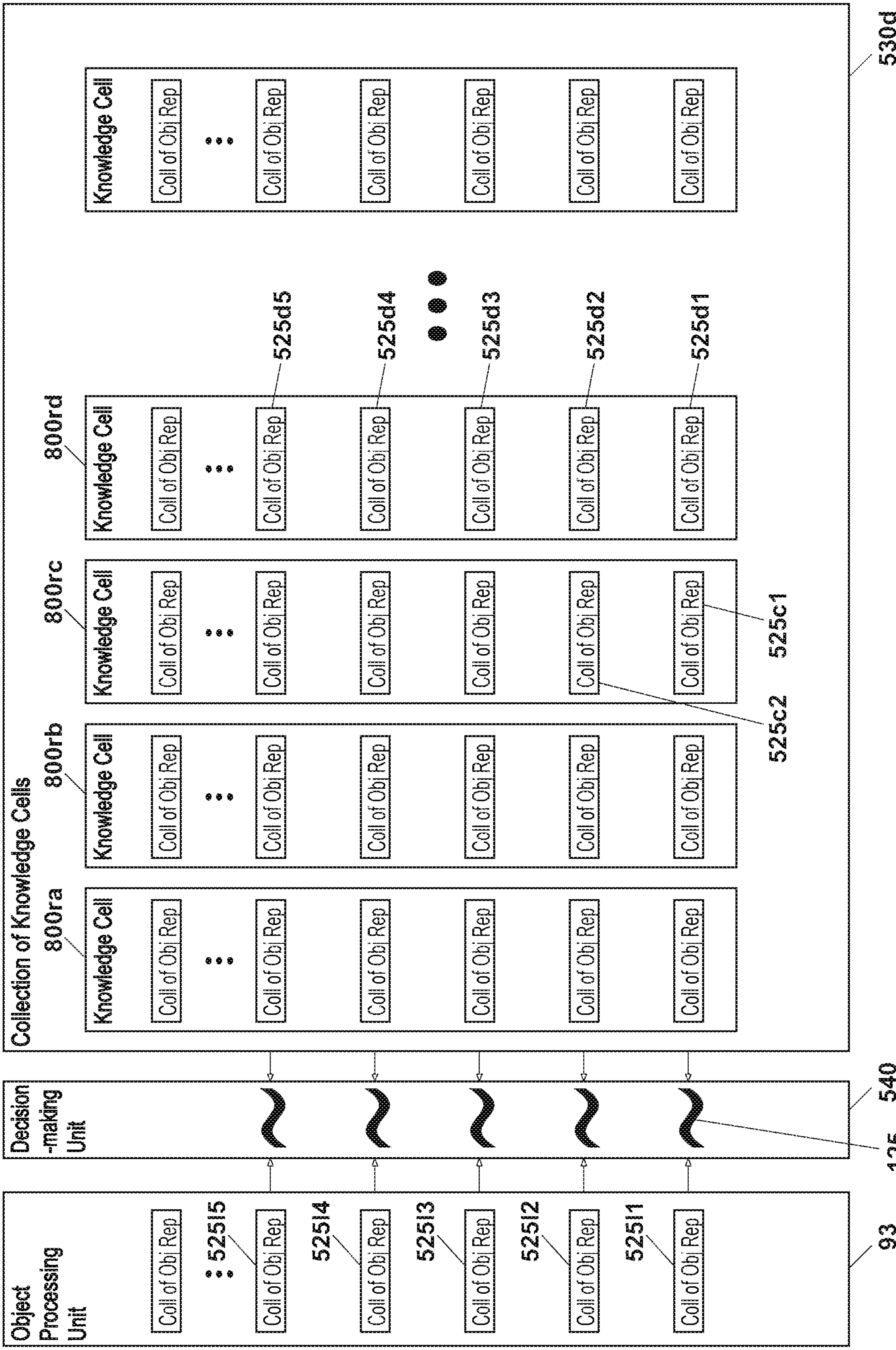

Object Processing Unit
Coll of Obj Rep
525l5
525l4
525l3
525l2
525l1
93
Decision -making Unit
540
125
Collection of Knowledge Cells
Knowledge Cell
800ra
800rb
800rc
800rd
525c1
525c2
525d1
525d2
525d3
525d4
525d5
530d

FIG. 26

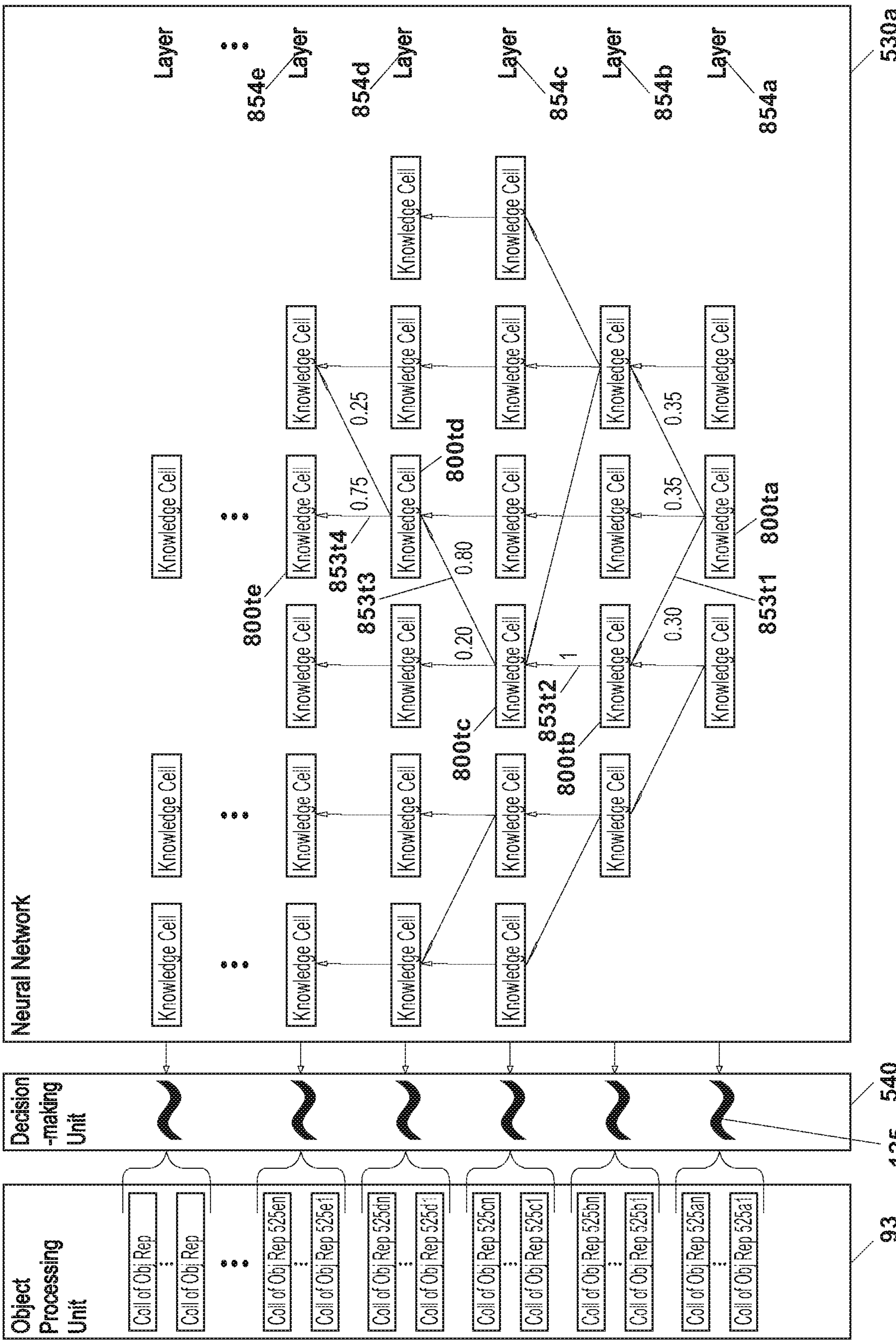
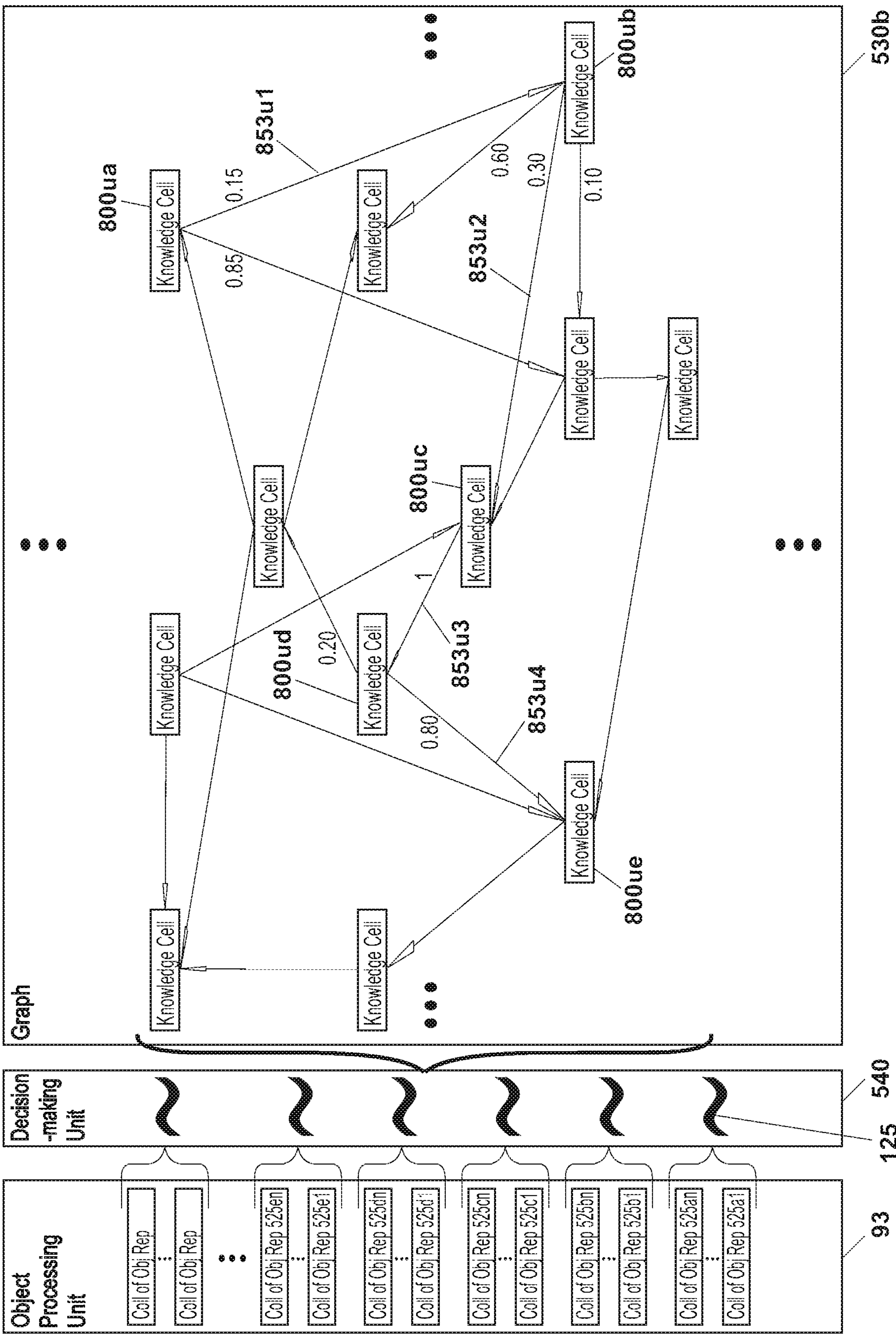

Neural Network
Knowledge Cell
Layer
854a
854b
854c
854d
854e
530a
800ta
800tb
800tc
800td
800te
853t1
853t2
853t3
853t4
0.20
0.80
0.75
0.25
0.35
0.35
0.30
1
Decision -making Unit
540
125
Object Processing Unit
Coll of Obj Rep
Coll of Obj Rep 525en
Coll of Obj Rep 525e1
Coll of Obj Rep 525dn
Coll of Obj Rep 525d1
Coll of Obj Rep 525cn
Coll of Obj Rep 525c1
Coll of Obj Rep 525bn
Coll of Obj Rep 525b1
Coll of Obj Rep 525an
Coll of Obj Rep 525a1
93

FIG. 27



Graph
800ua
Knowledge Cell
853u1
0.15
0.85
Knowledge Cell
0.60
0.30
Knowledge Cell
800ub
0.10
853u2
Knowledge Cell
Knowledge Cell
Knowledge Cell
800uc
Knowledge Cell
Knowledge Cell
1
800ud
0.20
Knowledge Cell
853u3
853u4
0.80
Knowledge Cell
800ue
Knowledge Cell
Knowledge Cell
530b
Decision
-making
Unit
540
125
Object
Processing
Unit
Coll of Obj Rep
Coll of Obj Rep
Coll of Obj Rep 525en
Coll of Obj Rep 525e1
Coll of Obj Rep 525dn
Coll of Obj Rep 525d1
Coll of Obj Rep 525cn
Coll of Obj Rep 525c1
Coll of Obj Rep 525bn
Coll of Obj Rep 525b1
Coll of Obj Rep 525an
Coll of Obj Rep 525a1
93

FIG. 28

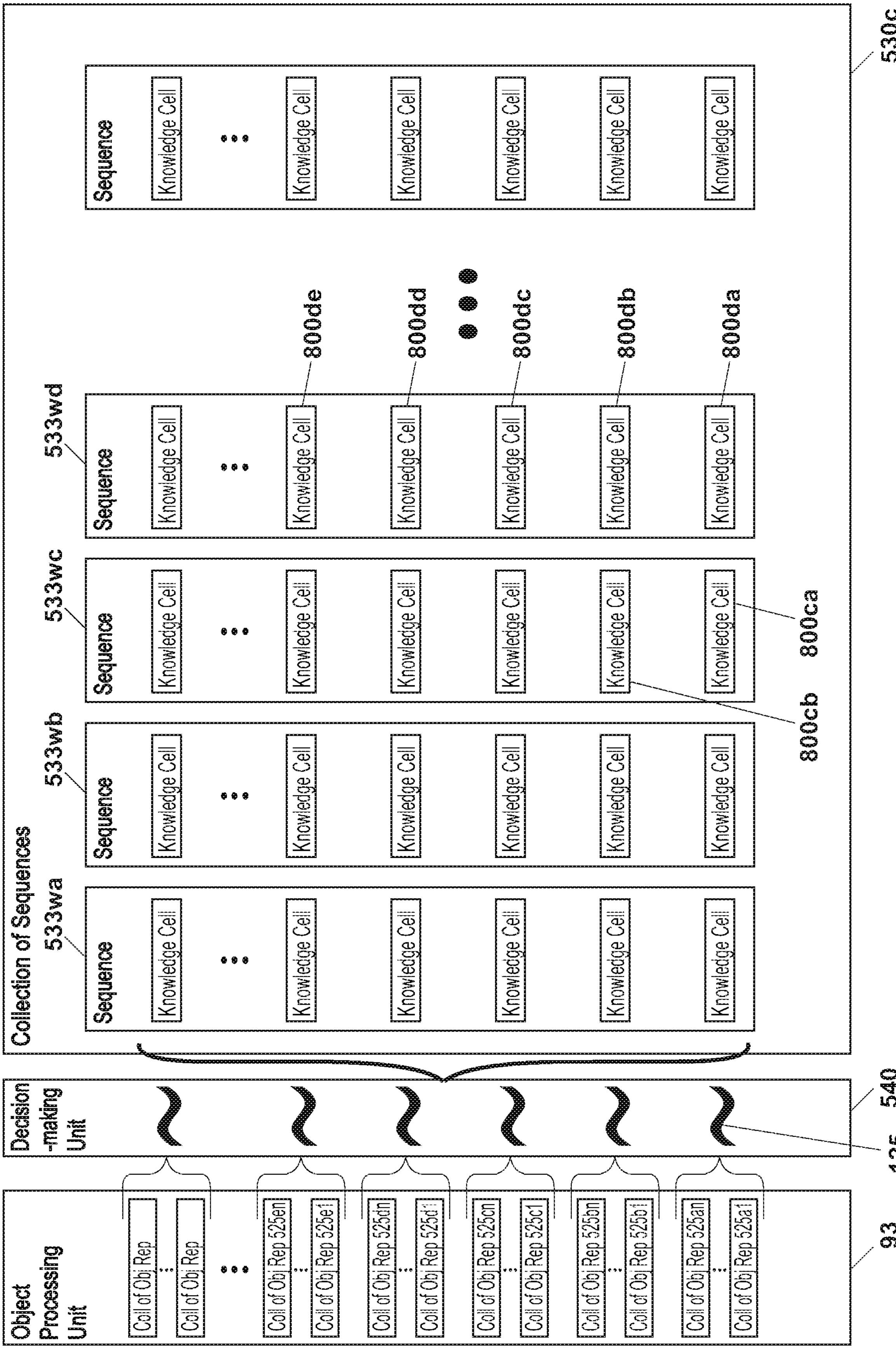
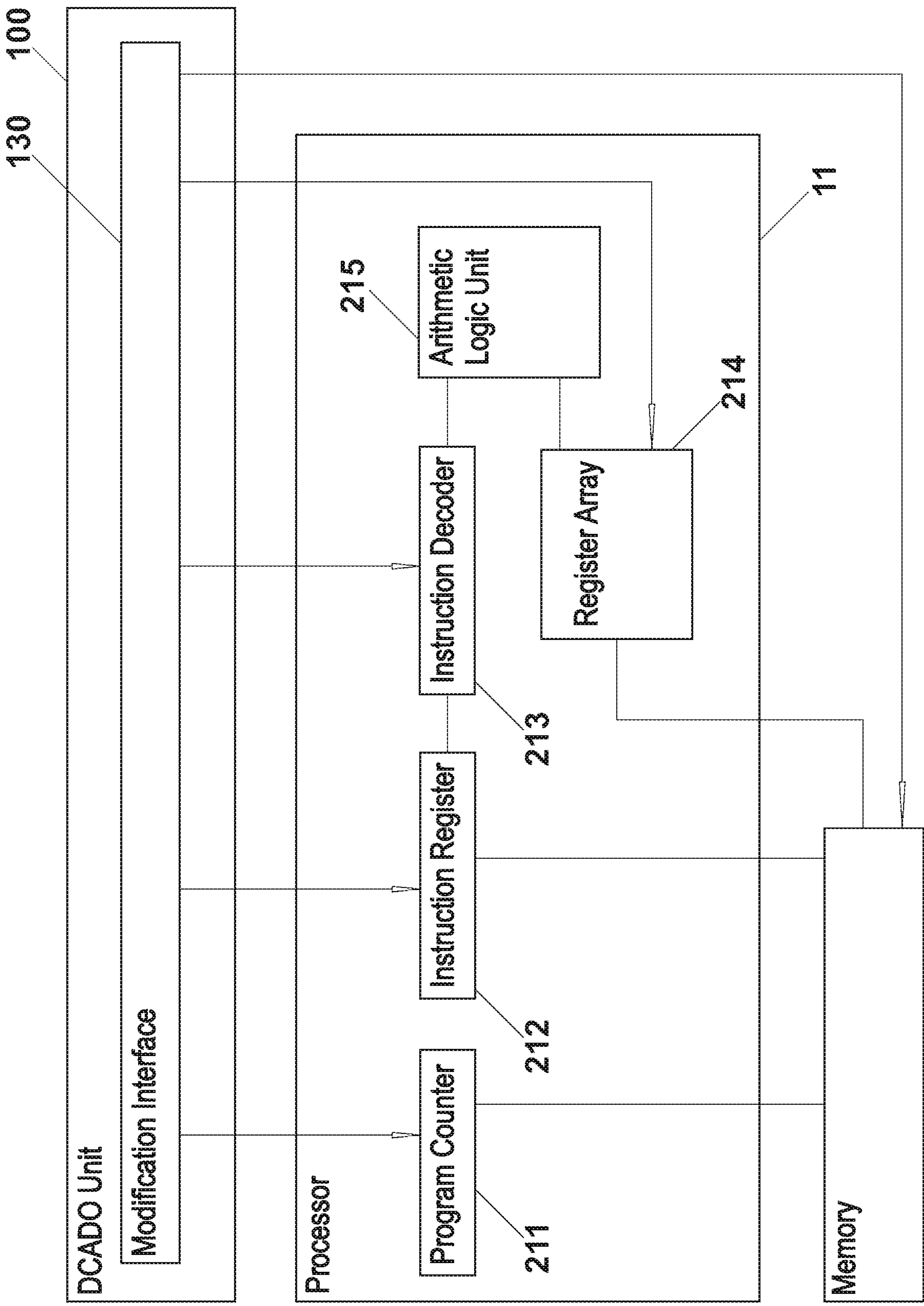

Object Processing Unit
Coll of Obj Rep
Coll of Obj Rep 525en
Coll of Obj Rep 525e1
Coll of Obj Rep 525dn
Coll of Obj Rep 525d1
Coll of Obj Rep 525cn
Coll of Obj Rep 525c1
Coll of Obj Rep 525bn
Coll of Obj Rep 525b1
Coll of Obj Rep 525an
Coll of Obj Rep 525a1
93
Decision -making Unit
540
125
Collection of Sequences
533wa
533wb
533wc
533wd
Sequence
Knowledge Cell
800ca
800cb
800da
800db
800dc
800dd
800de
530c

FIG. 29



100
130
DCADO Unit
Modification Interface
Processor
Program Counter
Instruction Register
Instruction Decoder
Arithmetic Logic Unit
Register Array
Memory
211
212
213
214
215
11
12

FIG. 30

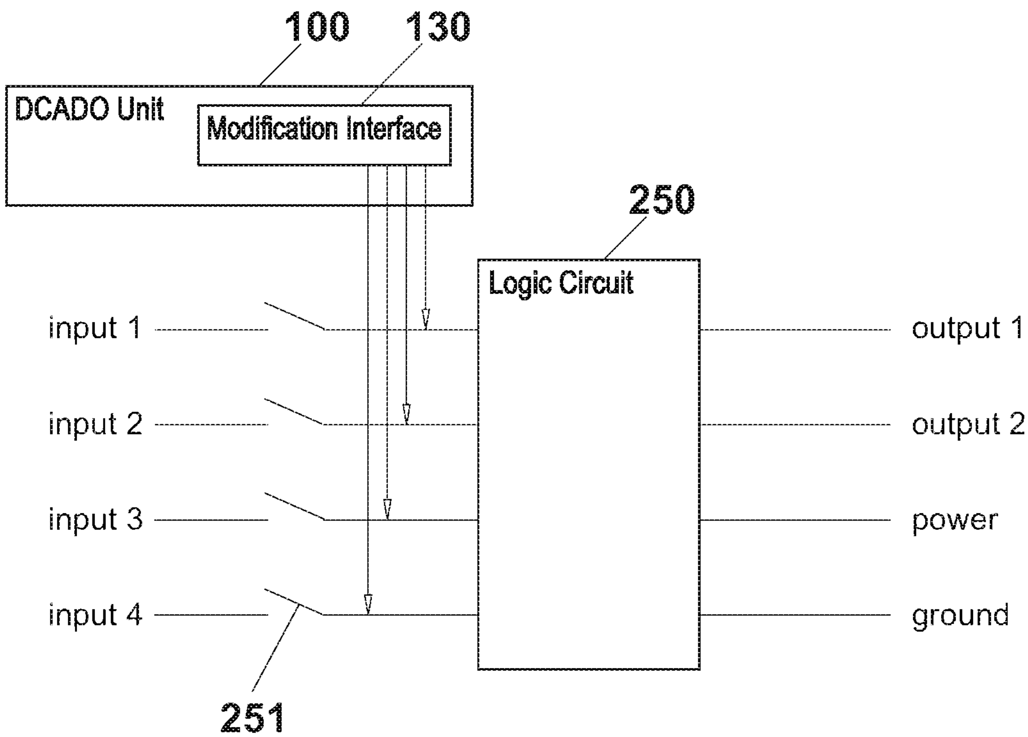


FIG. 31A

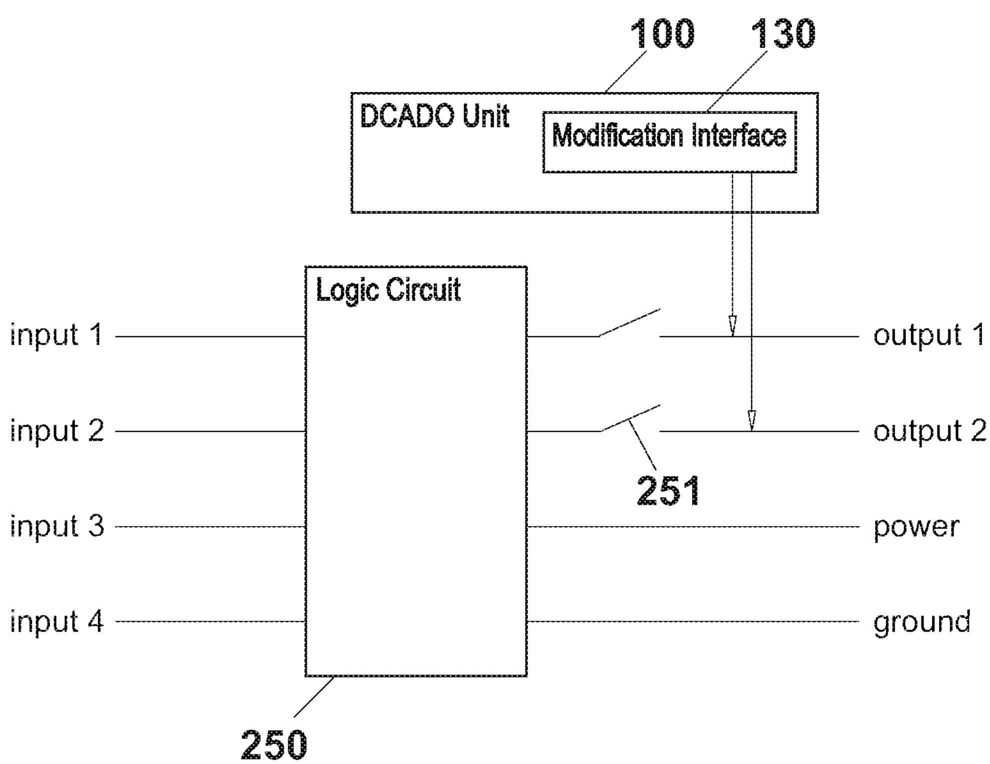


FIG. 31B

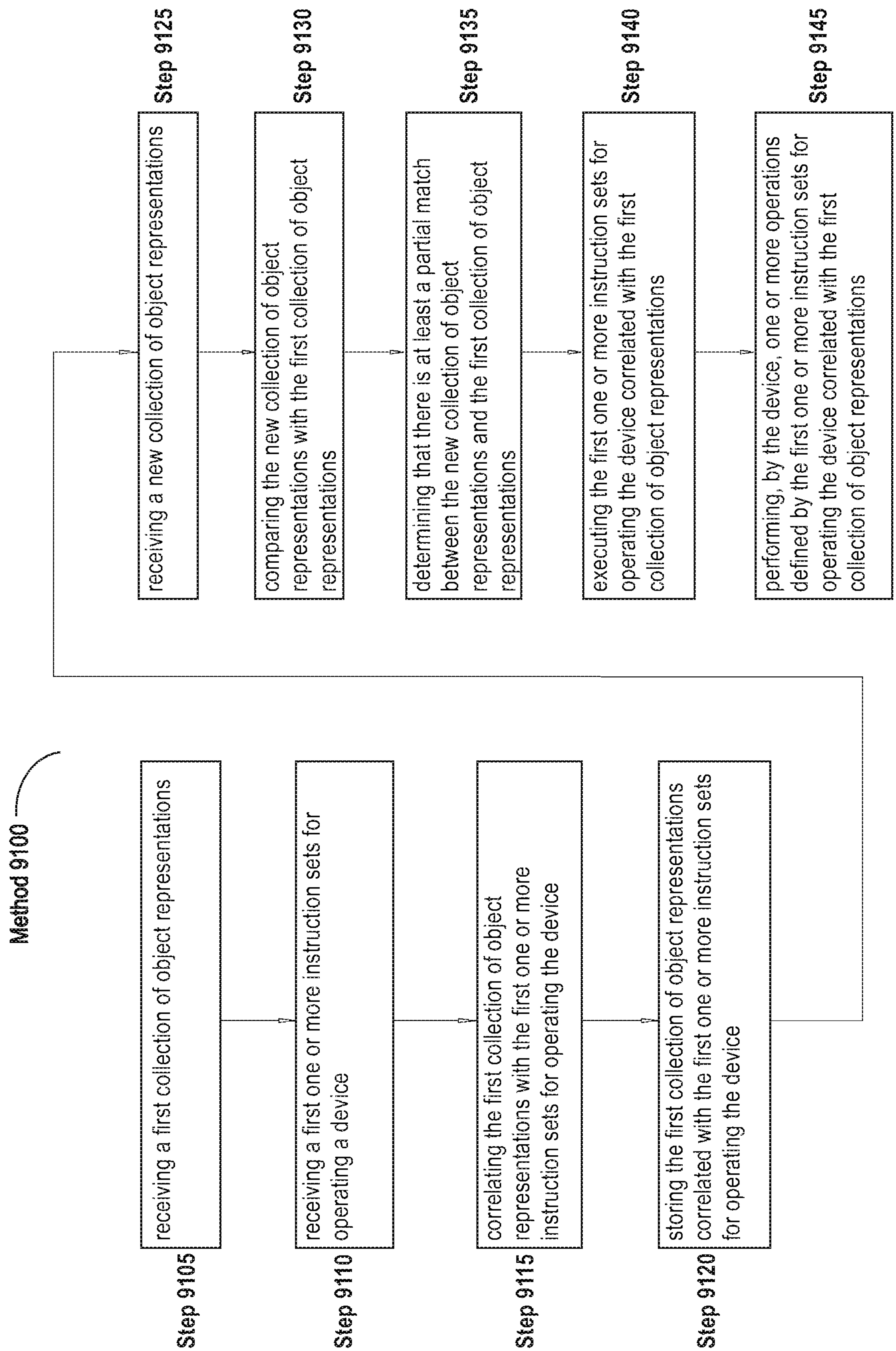

Method 9100
Step 9105
receiving a first collection of object representations
Step 9110
receiving a first one or more instruction sets for operating a device
Step 9115
correlating the first collection of object representations with the first one or more instruction sets for operating the device
Step 9120
storing the first collection of object representations correlated with the first one or more instruction sets for operating the device
Step 9125
receiving a new collection of object representations
Step 9130
comparing the new collection of object representations with the first collection of object representations
Step 9135
determining that there is at least a partial match between the new collection of object representations and the first collection of object representations
Step 9140
executing the first one or more instruction sets for operating the device correlated with the first collection of object representations
Step 9145
performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations

FIG. 32

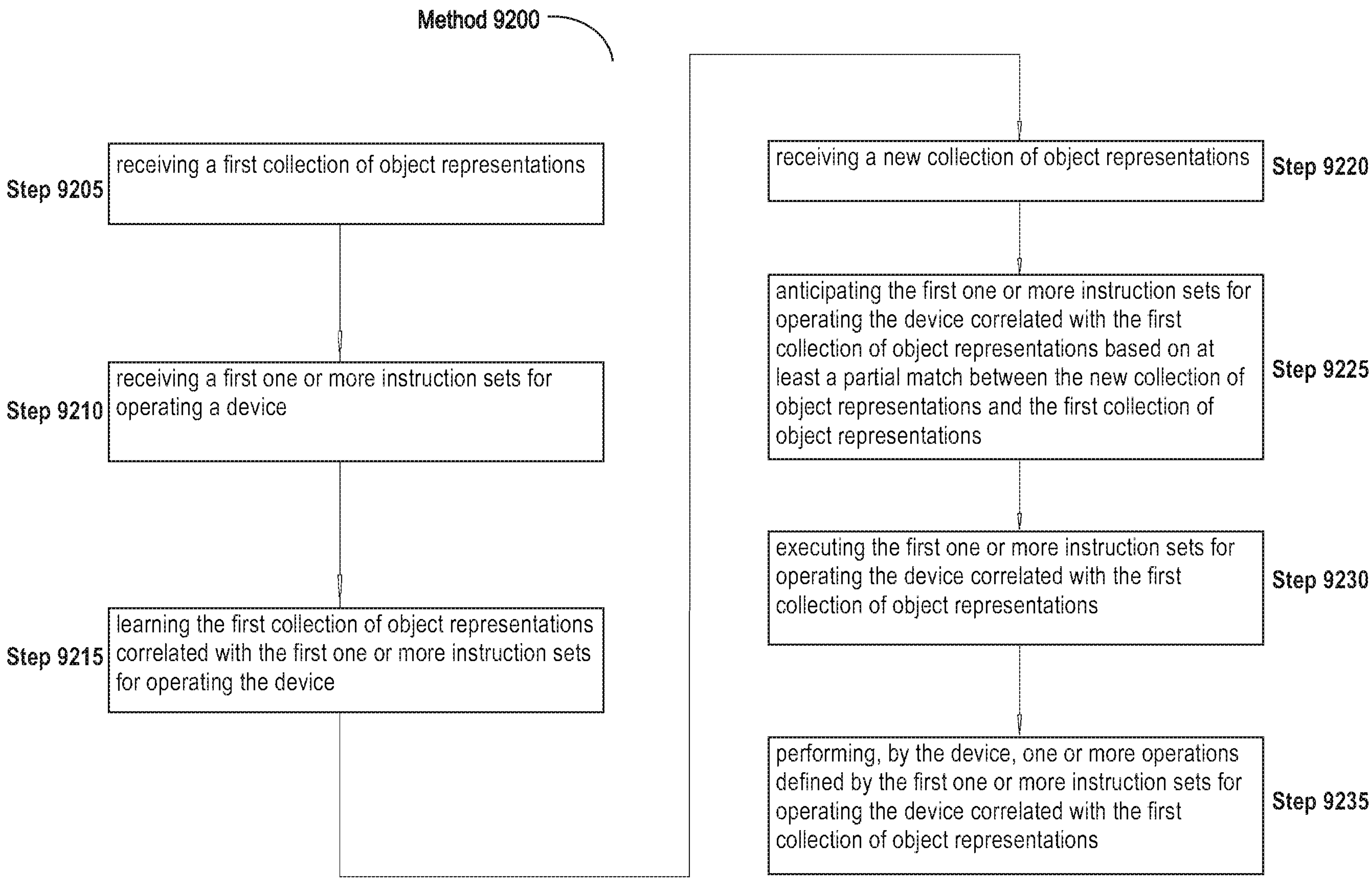
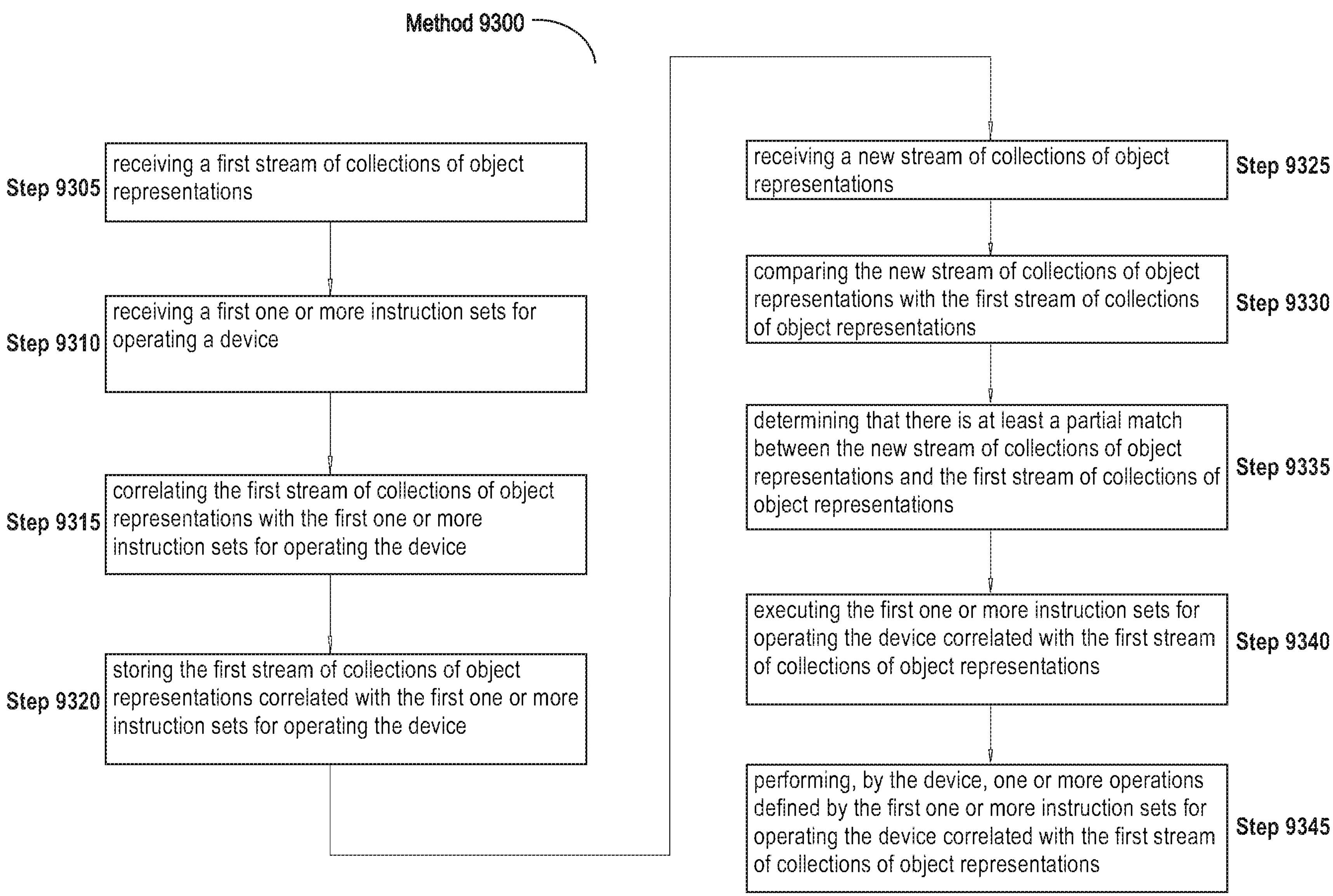

Method 9200
Step 9205
receiving a first collection of object representations
Step 9210
receiving a first one or more instruction sets for operating a device
Step 9215
learning the first collection of object representations correlated with the first one or more instruction sets for operating the device
Step 9220
receiving a new collection of object representations
Step 9225
anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations
Step 9230
executing the first one or more instruction sets for operating the device correlated with the first collection of object representations
Step 9235
performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations

FIG. 33



Method 9300
Step 9305
receiving a first stream of collections of object representations
Step 9310
receiving a first one or more instruction sets for operating a device
Step 9315
correlating the first stream of collections of object representations with the first one or more instruction sets for operating the device
Step 9320
storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device
Step 9325
receiving a new stream of collections of object representations
Step 9330
comparing the new stream of collections of object representations with the first stream of collections of object representations
Step 9335
determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations
Step 9340
executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations
Step 9345
performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations

FIG. 34

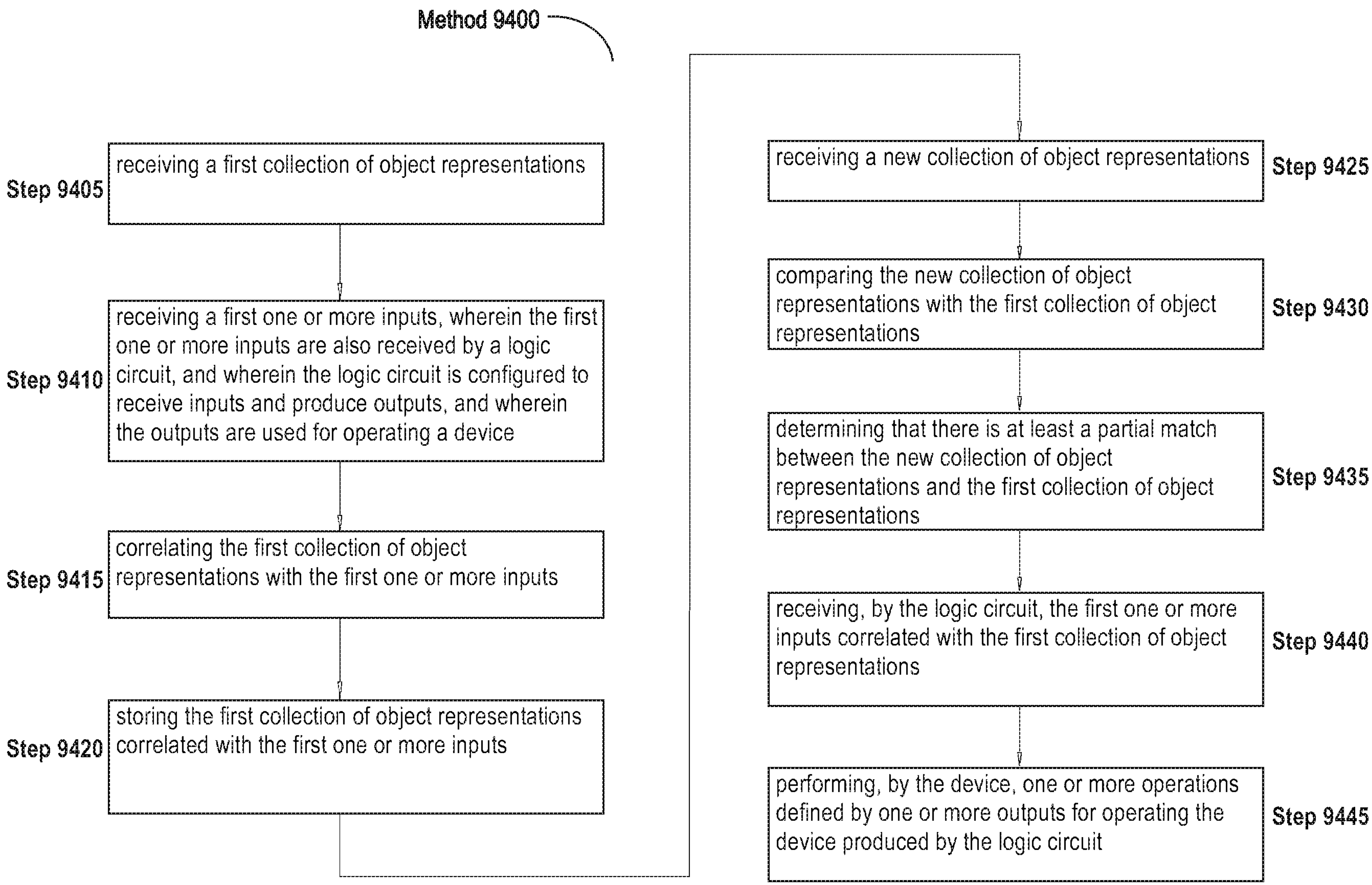

Method 9400
Step 9405
receiving a first collection of object representations
Step 9410
receiving a first one or more inputs, wherein the first one or more inputs are also received by a logic circuit, and wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device
Step 9415
correlating the first collection of object representations with the first one or more inputs
Step 9420
storing the first collection of object representations correlated with the first one or more inputs
Step 9425
receiving a new collection of object representations
Step 9430
comparing the new collection of object representations with the first collection of object representations
Step 9435
determining that there is at least a partial match between the new collection of object representations and the first collection of object representations
Step 9440
receiving, by the logic circuit, the first one or more inputs correlated with the first collection of object representations
Step 9445
performing, by the device, one or more operations defined by one or more outputs for operating the device produced by the logic circuit

FIG. 35

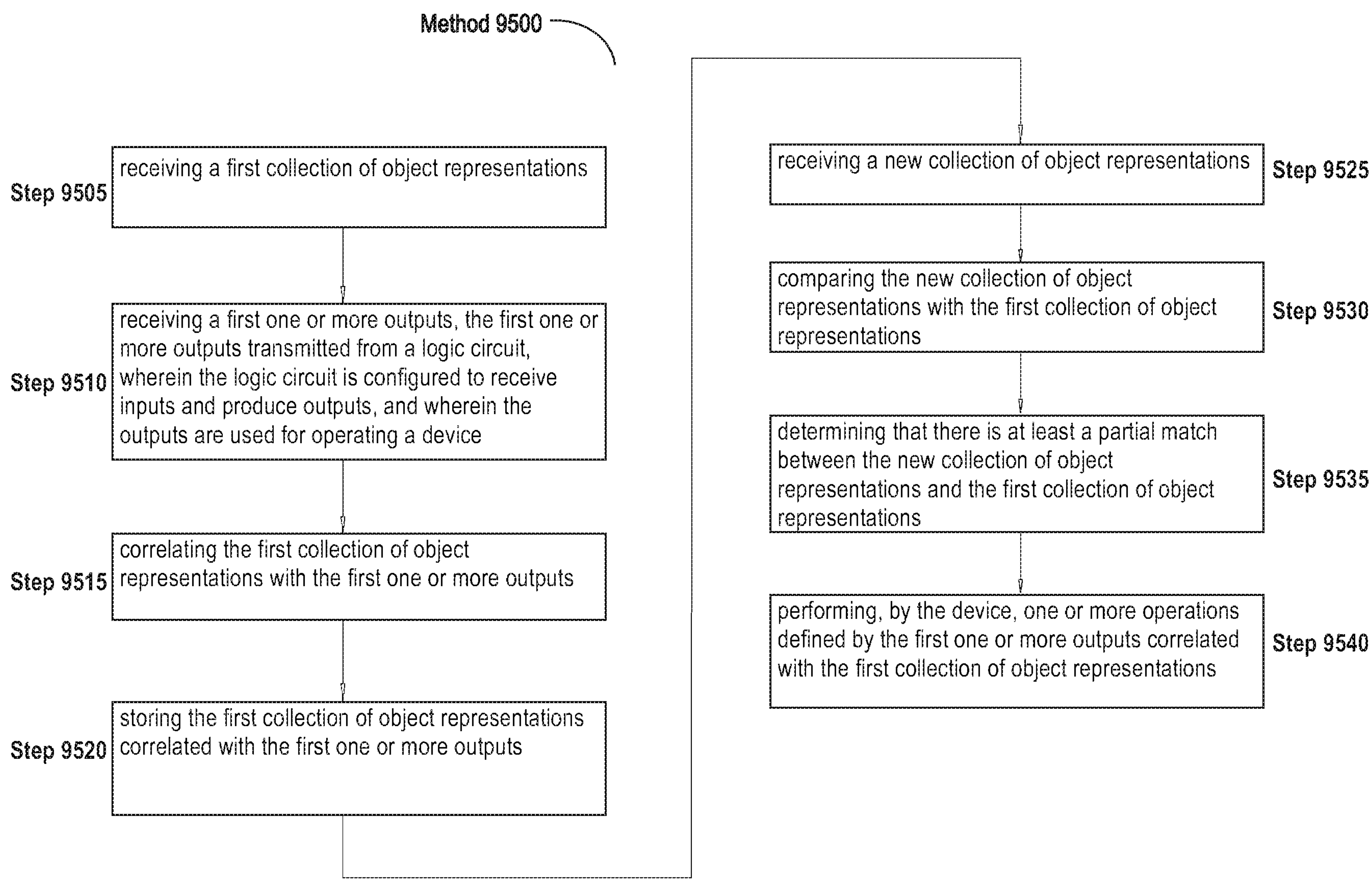

Method 9500
Step 9505
receiving a first collection of object representations
Step 9510
receiving a first one or more outputs, the first one or more outputs transmitted from a logic circuit, wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device
Step 9515
correlating the first collection of object representations with the first one or more outputs
Step 9520
storing the first collection of object representations correlated with the first one or more outputs
Step 9525
receiving a new collection of object representations
Step 9530
comparing the new collection of object representations with the first collection of object representations
Step 9535
determining that there is at least a partial match between the new collection of object representations and the first collection of object representations
Step 9540
performing, by the device, one or more operations defined by the first one or more outputs correlated with the first collection of object representations

FIG. 36

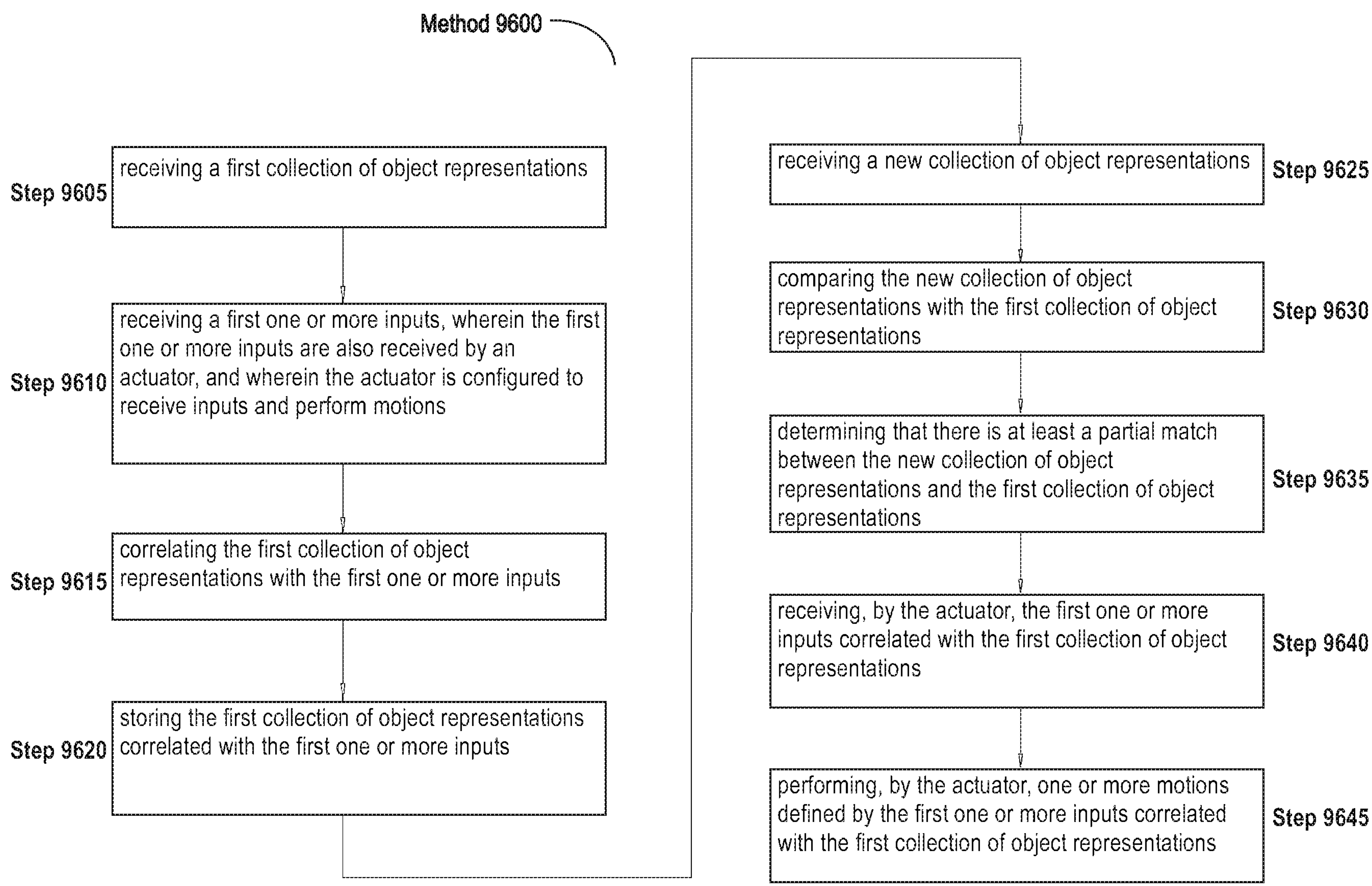

Method 9600
Step 9605
receiving a first collection of object representations
Step 9610
receiving a first one or more inputs, wherein the first one or more inputs are also received by an actuator, and wherein the actuator is configured to receive inputs and perform motions
Step 9615
correlating the first collection of object representations with the first one or more inputs
Step 9620
storing the first collection of object representations correlated with the first one or more inputs
Step 9625
receiving a new collection of object representations
Step 9630
comparing the new collection of object representations with the first collection of object representations
Step 9635
determining that there is at least a partial match between the new collection of object representations and the first collection of object representations
Step 9640
receiving, by the actuator, the first one or more inputs correlated with the first collection of object representations
Step 9645
performing, by the actuator, one or more motions defined by the first one or more inputs correlated with the first collection of object representations

FIG. 37

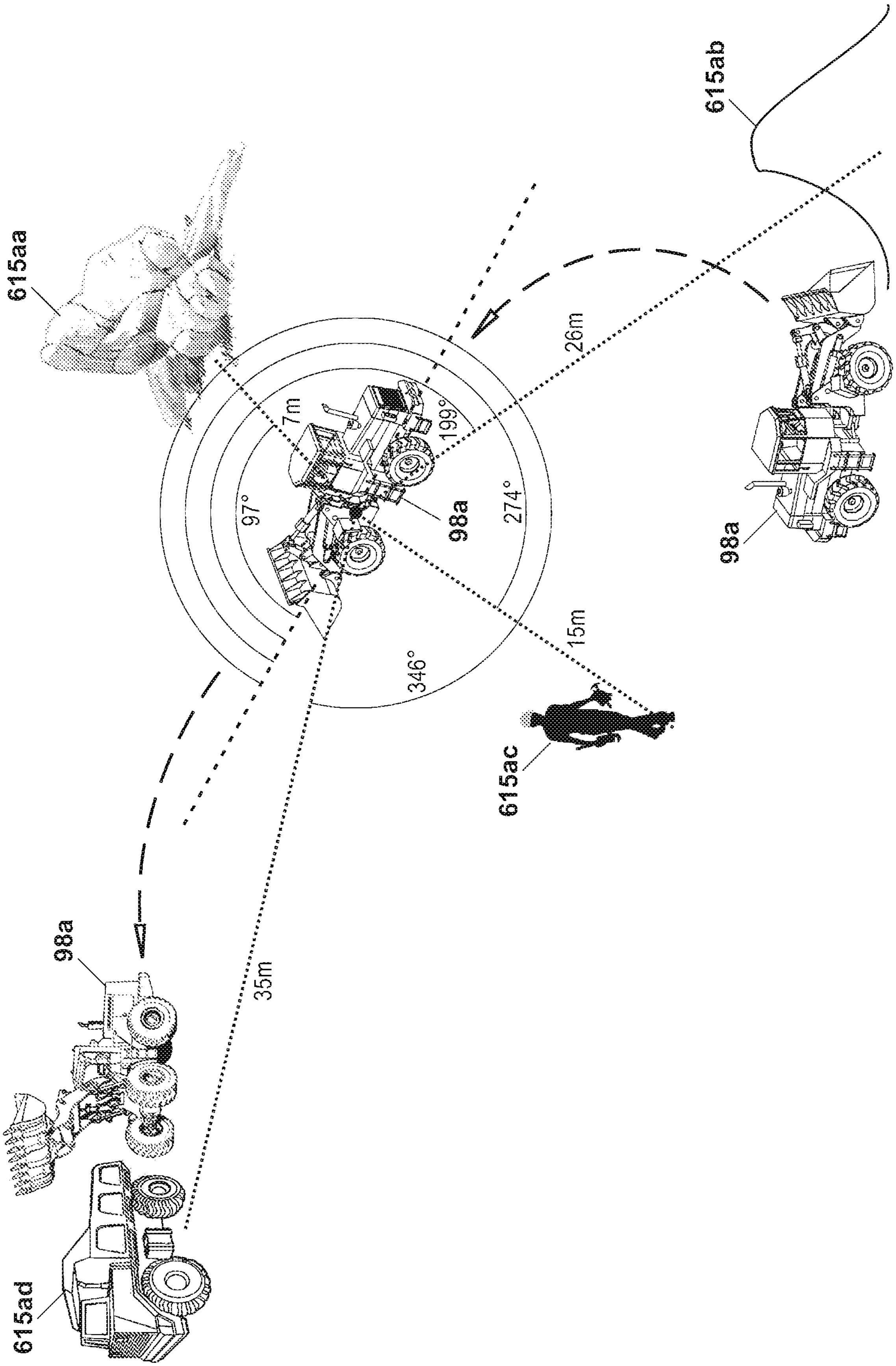

615aa
615ab
26m
7m
199°
97°
274°
98a
98a
15m
346°
615ac
98a
35m
615ad

FIG. 38

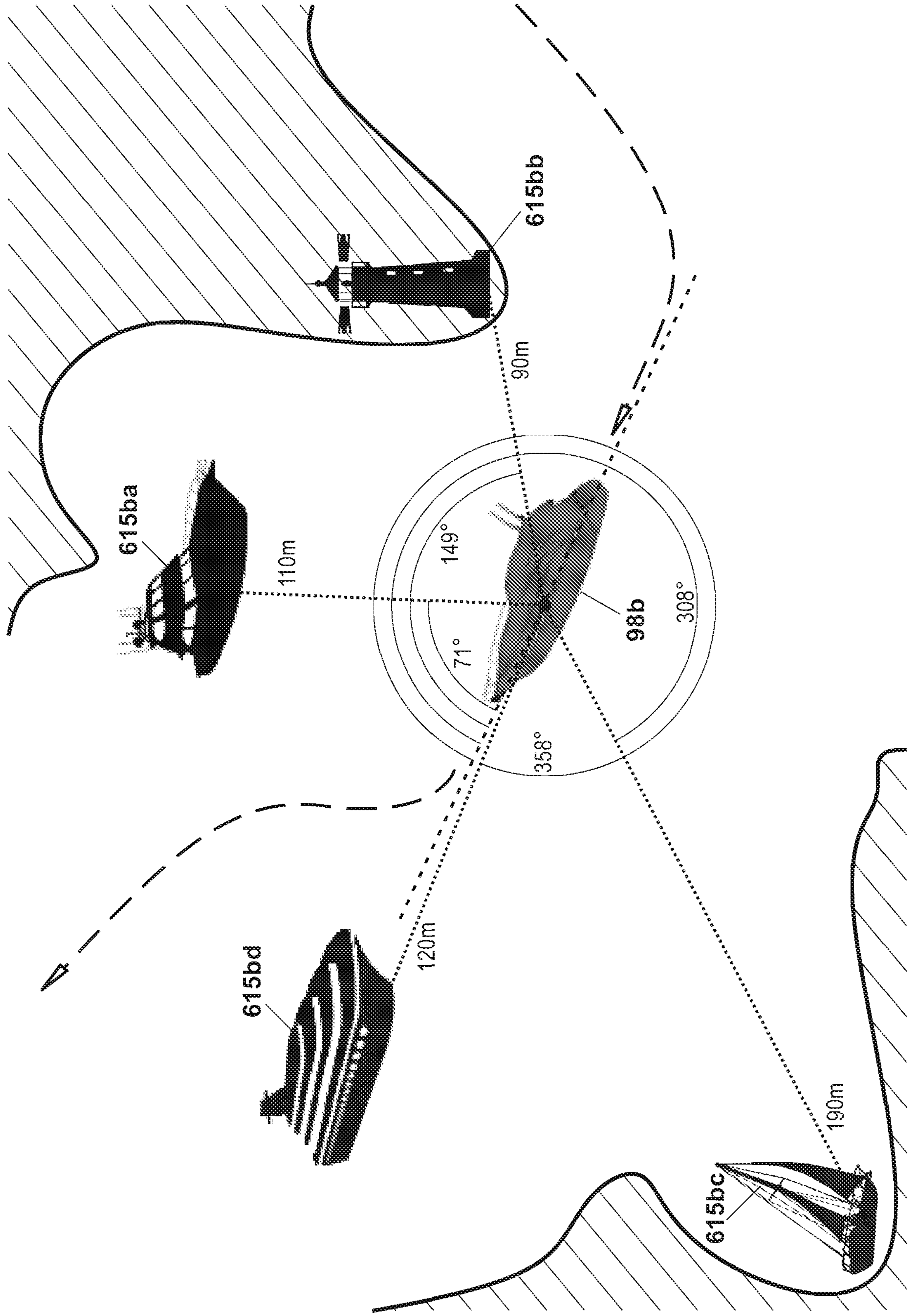


FIG. 39

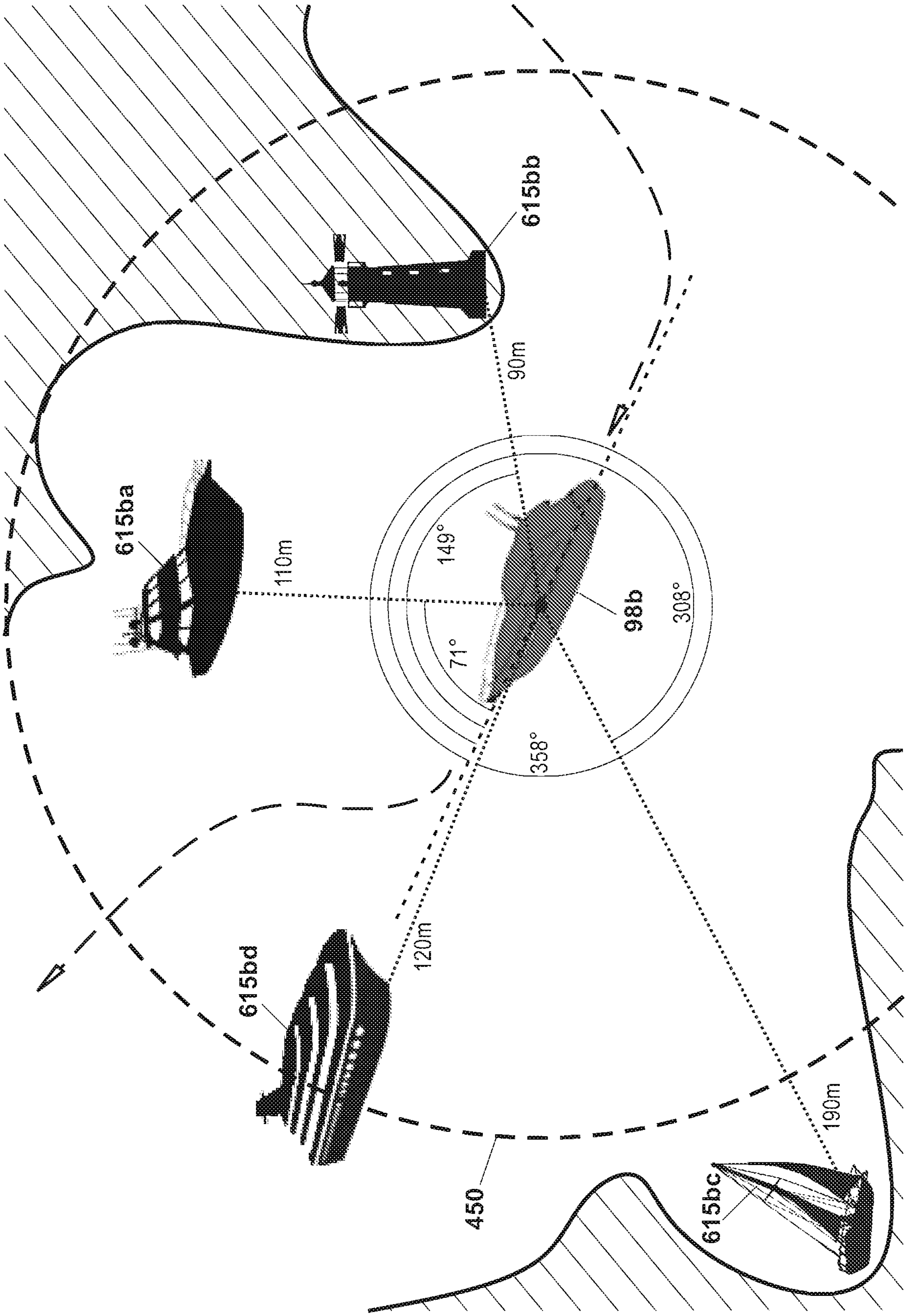

615bb
90m
615ba
110m
149°
98b
308°
71°
358°
120m
615bd
450
615bc
190m

FIG. 40

Electronic Patent Application Fee Transmittal

Application Number:	
Filing Date:	
Title of Invention:	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION
First Named Inventor/Applicant Name:	Jasmin Cosic
Filer:	Jasmin Cosic
Attorney Docket Number:	

Filed as Small Entity

Filing Fees for Utility under 35 USC 111(a)

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Basic Filing:				
UTILITY FILING FEE (ELECTRONIC FILING)	4011	1	70	70
UTILITY SEARCH FEE	2111	1	300	300
UTILITY EXAMINATION FEE	2311	1	360	360
Pages:				
UTILITY APPL SIZE FEE PER 50 SHEETS >100	2081	1	200	200
Claims:				
Miscellaneous-Filing:				
Petition:				

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Patent-Appeals-and-Interference:				
Post-Allowance-and-Post-Issuance:				
Extension-of-Time:				
Miscellaneous:				
	Total in USD ($)			**930**

Electronic Acknowledgement Receipt

EFS ID:	27391591
Application Number:	15340991
International Application Number:	
Confirmation Number:	1993
Title of Invention:	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION
First Named Inventor/Applicant Name:	Jasmin Cosic
Customer Number:	116094
Filer:	Jasmin Cosic
Filer Authorized By:	
Attorney Docket Number:	
Receipt Date:	02-NOV-2016
Filing Date:	
Time Stamp:	02:17:15
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	yes
Payment Type	CARD
Payment was successfully received in RAM	$930
RAM confirmation Number	110216INTEFSW02222800
Deposit Account	
Authorized User	
The Director of the USPTO is hereby authorized to charge indicated fees and credit any overpayment as follows:	

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1	Abstract	ABSTRACT_flattened.pdf	220691 da3ae9f7d0b2f5ba2cf0ec8b463a23fc7d46 5c06	no	1
Warnings:					
Information:					
2	Application Data Sheet	ADS.pdf	1822777 9f1905405423b472d1600b2612af4322cdb b8438	no	8
Warnings:					
Information:					
3	Oath or Declaration filed	DECLARATION_flattened.pdf	80488 20389855346f595c777e59543f418bd69a3 e225b	no	1
Warnings:					
Information:					
4	Specification	SPECIFICATION_flatenned.pdf	11824929 7189a9698cb43429c7efec2330d9b2154bc 58743	no	152
Warnings:					
Information:					
5	Claims	CLAIMS_flattened.pdf	2974744 2fd110cc8195f4644b63a7b3c8a86585bcd6 51e5	no	8
Warnings:					
Information:					
6	Drawings-only black and white line drawings	DRAWINGS_flattened.pdf	1137891 0551a2e3e7754760ebd68a3a1041bf9c4a8 edd7e	no	40
Warnings:					
The page size in the PDF is too large. The pages should be 8.5 x 11 or A4. If this PDF is submitted, the pages will be resized upon entry into the Image File Wrapper and may affect subsequent processing					

Information:					
7	Fee Worksheet (SB06)	fee-info.pdf	36505 c5001ebf77638214a3b771f56d2dfb907a865a87	no	2
Warnings:					
Information:					
		Total Files Size (in bytes):	18098025		

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.

National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

DocCode - SCORE

SCORE Placeholder Sheet for IFW Content

Application Number: 15340991 Document Date: 11/02/2016

The presence of this form in the IFW record indicates that the following document type was received in electronic format on the date identified above. This content is stored in the SCORE database.

- Drawings – Other than Black and White Line Drawings

Since this was an electronic submission, there is no physical artifact folder, no artifact folder is recorded in PALM, and no paper documents or physical media exist. The TIFF images in the IFW record were created from the original documents that are stored in SCORE.

To access the documents in the SCORE database, refer to instructions below.

At the time of document entry (noted above):

- Examiners may access SCORE content via the eDAN interface.
- Other USPTO employees can bookmark the current SCORE URL (http://Score.uspto.gov/ScoreAccessWeb/).
- External customers may access SCORE content via the Public and Private PAIR interfaces.

Form Revision Date: September 30, 2013

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed
PTO/SB/08a (01-10)
Approved for use through 07/31/2012. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS Remove

Examiner Initial*	Cite No	Patent Number	Kind Code1	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	4370707		1983-01-25	Phillips , et al.	
	2	4730315		1988-03-08	Saito , et al.	
	3	4860203		1989-08-22	Corrigan , et al.	
	4	5602982		1997-02-11	Judd , et al.	
	5	6026234		2000-02-15	Hanson , et al.	
	6	6088731		2000-07-11	Kiraly , et al.	
	7	6106299		2000-08-22	Ackermann , et al.	
	8	6126330		2000-10-03	Knight	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	9	6314558		2001-11-06	Angel , et al.	
	10	6643842		2003-11-04	Angel , et al.	
	11	6728689		2004-04-27	Drissi , et al.	
	12	6735632		2004-05-11	Kiraly , et al.	
	13	6801912		2004-10-05	Moskowitz , et al.	
	14	6915105		2005-07-05	Masuda	
	15	6973446		2005-12-06	Mamitsuka , et al.	
	16	7017153		2006-03-21	Gouriou , et al.	
	17	7052277		2006-05-30	Kellman	
	18	7222127		2007-05-22	Bem , et al.	
	19	7240335		2007-07-03	Angel , et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	20	7249349		2007-07-24	Hundt , et al.	
	21	7424705		2008-09-09	Lewis , et al.	
	22	7478371		2009-01-13	Gove	
	23	7484205		2009-01-27	Venkatapathy	
	24	7721218		2010-05-18	Awe , et al.	
	25	7765537		2010-07-27	Havin , et al.	
	26	7797259		2010-09-14	Jiang , et al.	
	27	7840060		2010-11-23	Podilchuk	
	28	7925984		2011-04-12	Awe , et al.	
	29	7987144		2011-07-26	Drissi , et al.	
	30	8005828		2011-08-23	Buchner , et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	31	8019699		2011-09-13	Baxter	
	32	8078556		2011-12-13	Adi , et al.	
	33	8090669		2012-01-03	Shahani , et al.	
	34	8137112		2012-03-20	Woolf , et al.	
	35	8166463		2012-04-24	Gill , et al.	
	36	8195674		2012-06-05	Bem , et al.	
	37	8196119		2012-06-05	Gill , et al.	
	38	8244730		2012-08-14	Gupta	
	39	8251704		2012-08-28	Woolf , et al.	
	40	8266608		2012-09-11	Hecht , et al.	
	41	8364612		2013-01-29	Van Gael , et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	42	8386401		2013-02-26	Virkar , et al.	
	43	8397227		2013-03-12	Fan , et al.	
	44	8464225		2013-06-11	Greifeneder	
	45	8549359		2013-10-01	Zheng	
	46	8589414		2013-11-19	Waite , et al.	
	47	8655260		2014-02-18	Yueh , et al.	
	48	8667472		2014-03-04	Molinari	

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS Remove

Examiner Initial*	Cite No	Publication Number	Kind Code[1]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	20120167057		2012-06-28	Schmich; Christopher P. ; et al.	
	2	20120150773		2012-06-14	DiCorpo; Phillip ; et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	3	20080254429		2008-10-16	Woolf; Susan D. ; et al.	
	4	20040117771		2004-06-17	Venkatapathy, Ramanathan	
	5	20130238533		2013-09-12	VIRKAR; Hemant ; et al.	
	6	20060190930		2006-08-24	Hecht; Daniel M. ; et al.	
	7	20070006159		2007-01-04	Hecht; Daniel Michael ; et al.	
	8	20080281764		2008-11-13	Baxter; Jonathan	
	9	20100138370		2010-06-03	Wu; Wang ; et al.	
	10	20110218672		2011-09-08	Maisonnier; Bruno ; et al.	

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number3	Country Code2 i	Kind Code4	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T5
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T5
	1	About Event Tracing, retrieved from <URL: http://msdn.microsoft.com/en-us/library/aa363668(d=default,l=en-us, v=vs.85).aspx> on Jan 12, 2014, 2 pages	
	2	Add dynamic Java code to your application, retrieved from <URL: http://www.javaworld.com/article/2071777/design-patterns/add-dynamic-java-code-to-your-application.html> on May 13, 2014, 6 pages	
	3	BCEL, retrieved from <URL: http://commons.apache.org> on May 13, 2014, 2 pages	
	4	Artificial intelligence, retrieved from <URL: http://wikipedia.com> on May 21, 2014, 28 pages	
	5	Artificial neural network, retrieved from <URL: http://wikipedia.com> on May 21, 2014, 12 pages	
	6	Branch (computer science), retrieved from <URL: http://wikipedia.com> on May 6, 2014, 2 pages	
	7	Branch trace, retrieved from <URL: http://wikipedia.com> on May 28, 2014, 2 pages	
	8	C Function Call Conventions and the Stack, retrieved from <URL: http://www.csee.umbc.edu/~chang/cs313.s02/stack.shtml> on Jan 11, 2014, 4 pages	
	9	Call stack, retrieved from <URL: http://wikipedia.com> on Jan 11, 2014, 5 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	10	Program compilation and execution flow, retrieved from <URL: http://cs.stackexchange.com/questions/6187/program-compilation-and-execution-flow> on Jan 9, 2014, 2 pages	
	11	Control unit, retrieved from <URL: http://wikipedia.com> on May 25, 2014, 2 pages	
	12	Create dynamic applications with javax.tools, retrieved from <URL: http://www.ibm.com/developerworks/library/j-jcomp/> on May 13, 2014, 11 pages	
	13	Creating a Debugging and Profiling Agent with JVMTI, retrieved from <URL: http://www.oracle.com/technetwork/articles/java/jvmti-136367.html> on Jan 11, 2014, 18 pages	
	14	Decision tree learning, retrieved from <URL: http://wikipedia.com> on Jun 24, 2014, 5 pages	
	15	Dynamic Bytecode Instrumentation, retrieved from <URL: http://www.drdobbs.com/tools/dynamic-bytecode-instrumentation/184406433> on Apr 26, 2014, 6 pages	
	16	Dynamic programming language, retrieved from <URL: http://wikipedia.com> on May 7, 2014, 4 pages	
	17	Dynamic recompilation, retrieved from <URL: http://wikipedia.com> on May 19, 2014, 4 pages	
	18	eval(), retrieved from <URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/eval> on May 7, 2014, 13 pages	
	19	Function, retrieved from <URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Function> on May 7, 2014, 11 pages	
	20	FunctionEnter2 Function, retrieved from <URL: https://msdn.microsoft.com/en-us/library/aa964981(d=default,l=en-us,v=vs.110).aspx> on Mar 19, 2014, 2 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	21	FunctionLeave2 Function, retrieved from <URL: https://msdn.microsoft.com/en-us/library/aa964942(d=default,l=en-us,v=vs.110).aspx> on Mar 19, 2014, 1 pages	
	22	Heuristic, retrieved from <URL: http://wikipedia.com> on May 22, 2014, 5 pages	
	23	How to: Configure Trace Switches, retrieved from <URL: https://msdn.microsoft.com/en-us/library/vstudio/t06xyy08(d=default,l=en-us,v=vs.100).aspx> on Mar 19, 2014, 3 pages	
	24	Indirect branch, retrieved from <URL: http://wikipedia.com> on May 5, 2014, 1 pages	
	25	Instruction cycle, retrieved from <URL: http://wikipedia.com> on Apr 30, 2014, 2 pages	
	26	Instruction set, retrieved from <URL: http://wikipedia.com> on Apr 29, 2014, 6 pages	
	27	Interrupt, retrieved from <URL: http://wikipedia.com> on May 15, 2014, 5 pages	
	28	Introduction to Instrumentation and Tracing, retrieved from <URL: http://msdn.microsoft.com/en-us/library/aa983649(d=default,l=en-us,v=vs.71).aspx> on Jan 10, 2014, 2 pages	
	29	Java bytecode, retrieved from <URL: http://wikipedia.com> on May 27, 2014, 4 pages	
	30	Package java.util.logging, retrieved from <URL: http://docs.oracle.com/javase/1.5.0/docs/api/java/util/logging/package-summary.html> on Mar 19, 2014, 2 pages	
	31	javac - Java programming language compiler, retrieved from <URL: http://docs.oracle.com/javase/7/docs/technotes/tools/windows/javac.html> on May 13, 2014, 12 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	32	Interface JavaCompiler, retrieved from <URL: http://wikipedia.com> on May 13, 2014, 4 pages	
	33	Javassist, retrieved from <URL: http://wikipedia.com> on May 13, 2014, 2 pages	
	34	Just-in-time compilation, retrieved from <URL: http://wikipedia.com> on May 19, 2014, 5 pages	
	35	JVM Tool Interface, retrieved from <URL: http://docs.oracle.com/javase/7/docs/platform/jvmti/jvmti.html> on Apr 26, 2014, 127 pages	
	36	Machine code, retrieved from <URL: http://wikipedia.com> on Apr 30, 2014, 3 pages	
	37	Metaprogramming, retrieved from <URL: http://wikipedia.com> on May 8, 2014, 3 pages	
	38	Microsoft.VisualBasic.Logging Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/ms128122(d=default,l=en-us,v=vs.110).aspx> on Jan 13, 2014, 1 pages	
	39	Pin (computer program), retrieved from <URL: http://wikipedia.com> on May 6, 2014, 3 pages	
	40	Processor register, retrieved from <URL: http://wikipedia.com> on Apr 29, 2014, 4 pages	
	41	Profiling (computer programming), retrieved from <URL: http://wikipedia.com> on Jan 10, 2014, 4 pages	
	42	Profiling Overview, retrieved from <URL: http://msdn.microsoft.com/en-us/library/bb384493(d=default,l=en-us,v=vs.110).aspx> on Jan 12, 2014, 6 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	43	Program counter, retrieved from <URL: http://wikipedia.com> on Apr 24, 2014, 2 pages	
	44	ptrace, retrieved from <URL: http://wikipedia.com> on Apr 24, 2014, 2 pages	
	45	Reflection (computer programming), retrieved from <URL: http://wikipedia.com> on May 7, 2014, 5 pages	
	46	Class Runtime, retrieved from <URL: http://docs.oracle.com/javase/7/docs/api/java/lang/Runtime.html> on Jan 10, 2014, 10 pages	
	47	Self-modifying code, retrieved from <URL: http://wikipedia.com> on Jan 12, 2014, 6 pages	
	48	Subjective logic, retrieved from <URL: http://wikipedia.com> on May 22, 2014, 5 pages	
	49	System.CodeDom.Compiler Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/z6b99ydt(d=default,l=en-us,v=vs.110).aspx> on May 13, 2014, 2 pages	
	50	System.Reflection.Emit Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/xd5fw18y(d=default,l=en-us,v=vs.110).aspx> on May 13, 2014, 2 pages	

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature		Date Considered	

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Jasmin Cosic/	Date (YYYY-MM-DD)	2016-11-10
Name/Print	Jasmin Cosic	Registration Number	

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS

Remove

Examiner Initial*	Cite No	Patent Number	Kind Code[1]	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1					

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS

Remove

Examiner Initial*	Cite No	Publication Number	Kind Code[1]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1					

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS

Remove

Examiner Initial*	Cite No	Foreign Document Number[3]	Country Code[2] i	Kind Code[4]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T[5]
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS

Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T[5]

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	1	Tracing (software), retrieved from <URL: http://wikipedia.com> on Jan 10, 2014, 3 pages	
	2	Tree (data structure), retrieved from <URL: http://wikipedia.com> on Jun 24, 2014, 6 pages	
	3	PTRACE(2), retrieved from <URL: http://unixhelp.ed.ac.uk/CGI/man-cgi?ptrace> on Mar 19, 2014, 5 pages	
	4	Wevtutil, retrieved from <URL: http://technet.microsoft.com/en-us/library/cc732848(d=default,l=en-us,v=ws.11).aspx> on Apr 28, 2014, 5 pages	
	5	Intel Processor Trace, retrieved from <URL: https://software.intel.com/en-us/blogs/2013/09/18/processor-tracing> on Apr 28, 2014, 3 pages	
	6	YOUNGHOON JUNG, JAVA DYNAMICS Reflection and a lot more, Oct 10, 2012, 55 pages, Columbia University	
	7	AMITABH SRIVASTAVA, ALAN EUSTACE, ATOM A System for Building Customized Program Analysis Tools, May 3, 2004, 12 pages	
	8	MATHEW SMITHSON, KAPIL ANAND, APARNA KOTHA, KHALED ELWAZEER, NATHAN GILES, RAJEEV BARUA, Binary Rewriting without Relocation Information, Nov 10, 2010, 11 pages, University of Maryland	
	9	MAREK OLSZEWSKI, KEIR MIERTE, ADAM CZAJKOWSKI, ANGELA DEMLE BROWN, JIT Instrumentation - A Novel Approach To Dynamically Instrument Operating Systems, Feb 12, 2007, 14 pages, University of Toronto	

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature		Date Considered	

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Jasmin Cosic/	Date (YYYY-MM-DD)	2016-11-10
Name/Print	Jasmin Cosic	Registration Number	

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS

Remove

Examiner Initial*	Cite No	Patent Number	Kind Code[1]	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1					

If you wish to add additional U.S. Patent citation information please click the Add button.

Add

U.S.PATENT APPLICATION PUBLICATIONS

Remove

Examiner Initial*	Cite No	Publication Number	Kind Code[1]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	20030026588		2003-02-06	Elder, James H. ; et al.	
	2	20040249774		2004-12-09	Caid, William R. ; et al.	
	3	20040267521		2004-12-30	Cutler, Ross ; et al.	
	4	20050245303		2005-11-03	Graepel, Kurt Hartwig ; et al.	
	5	20060047612		2006-03-02	Stanley; Kenneth O. ; et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	6	20070058856		2007-03-15	Boregowda; Lokesh R. ; et al.	
	7	20070106633		2007-05-10	Reiner; Bruce	
	8	20080144893		2008-06-19	GUO; Chun Biao ; et al.	
	9	20080288259		2008-11-20	Chambers; Robert L. ; et al.	
	10	20090110061		2009-04-30	LEE; Kang Yong ; et al.	
	11	20090131152		2009-05-21	Busse; Martin A.	
	12	20090141969		2009-06-04	Yu; Kai ; et al.	
	13	20090222388		2009-09-30	Hua; Wei ; et al.	
	14	20090324010		2009-12-31	Hou; Billy	
	15	20100033780		2010-02-11	Gitter; Kurt	
	16	20100063949		2010-03-11	EATON; JOHN ERIC ; et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	17	20100241595		2010-09-23	Felsher; David Paul	
	18	20100278420		2010-11-04	Shet; Vinay Damodar ; et al.	
	19	20110030031		2011-02-03	Lussier; Paul ; et al.	
	20	20130156345		2013-06-20	Shmunk; Dmitry Valerievich	
	21	20130159021		2013-06-20	Felsher; David Paul	
	22	20130278631		2013-10-24	Border; John N. ; et al.	
	23	20140052717		2014-02-20	FRANKS; Kasian ; et al.	
	24	20140161250		2014-06-12	Leeds; Richard ; et al.	
	25	20140177946		2014-06-26	LIM; Kil-Taek ; et al.	
	26	20140207580		2014-06-24	Minnis; William Charles ; et al.	
	27	20150039304		2015-02-05	Wein; Ron	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	28	20150055821		2015-02-26	Fotland; David Allen	

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number[3]	Country Code[2] i	Kind Code[4]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T[5]
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T[5]
	1	ABOUT | OpenCV, retrieved from <URL: http://opencv.org/about.html> on Dec 13, 2014, 1 pages	
	2	Animetrics, Inc. | 3D Facial Recognition, retrieved from <URL: http://animetrics.com/> on Dec 13, 2014, 2 pages	
	3	Bag-of-words model, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	4	Beier–Neely morphing algorithm, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 1 pages	
	5	Computer vision, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 7 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	6	Convolutional neural network, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 5 pages	
	7	Decimation (signal processing), retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 3 pages	
	8	Digital image processing, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 3 pages	
	9	Dissolve (filmmaking), retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 2 pages	
	10	Facial recognition system, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 6 pages	
	11	Feature detection (computer vision), retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 3 pages	
	12	Feature extraction, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	13	Gesture recognition, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 5 pages	
	14	GrabCut, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 1 pages	
	15	Image processing, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	16	Image segmentation, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 12 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	17	Image warping, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 2 pages	
	18	Inbetweening, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 2 pages	
	19	Interpolation, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 5 pages	
	20	Language model, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 4 pages	
	21	List of speech recognition software, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 4 pages	
	22	Livewire Segmentation Technique, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 2 pages	
	23	Morphing, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 2 pages	
	24	Motion estimation, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	25	Muse: Face Morph || Mesh Warping, retrieved from <URL: http://alexwolfe.blogspot.com/2011/10/face-morph-mesh-warping.html> on Nov 19, 2015, 2 pages	
	26	n-gram, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 5 pages	
	27	Optical flow, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 4 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	28	Outline of object recognition, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 7 pages	
	29	Phonotactics, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 3 pages	
	30	Recurrent neural network, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 5 pages	
	31	Sample rate conversion, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 2 pages	
	32	Simple interactive object extraction, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	33	Speech recognition, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 12 pages	
	34	Speech segmentation, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 4 pages	
	35	Welcome to recognize-speech.com, retrieved from <URL: http://recognize-speech.com/> on Oct 18, 2015, 1 pages	
	36	Introduction Speech, retrieved from <URL: http://recognize-speech.com/speech> on Oct 18, 2015, 1 pages	
	37	Preprocessing, retrieved from <URL: http://recognize-speech.com/preprocessing> on Oct 18, 2015, 4 pages	
	38	Feature Extraction, retrieved from <URL: http://recognize-speech.com/feature-extraction> on Oct 18, 2015, 3 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	39	Acoustic model, retrieved from <URL: http://recognize-speech.com/acoustic-model> on Oct 18, 2015, 2 pages	
	40	Video content analysis, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	41	Video tracking, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 2 pages	
	42	Andrej Karpathy, Thomas Leung, George Toderici, Rahul Sukthankar, Sanketh Shetty, Li Fei-Fei, Large-scale Video Classification with Convolutional Neural Networks, Apr 14, 2014, 8 pages, Stanford University	
	43	Karen Simonyan, Andrew Zisserman, Two-Stream Convolutional Networks for Action Recognition in Videos, Nov 13, 2014, 11 pages, University of Oxford	

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature		Date Considered	

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Jasmin Cosic/	Date (YYYY-MM-DD)	2016-11-10
Name/Print	Jasmin Cosic	Registration Number	

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

PTO/SB/08a (01-10)
Approved for use through 07/31/2012. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS Remove

Examiner Initial*	Cite No	Patent Number	Kind Code1	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	7113946		2006-09-26	Cosic	
	2	7117225		2006-10-03	Cosic	
	3	8335805		2012-12-18	Cosic	
	4	8417740		2013-04-09	Cosic	
	5	8572035		2013-10-29	Cosic	
	6	8655900		2014-02-18	Cosic	
	7	9047324		2015-06-02	Cosic	
	8	9282309		2016-03-08	Cosic	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	9	9298749		2016-03-29	Cosic	
	10	9367806		2016-06-14	Cosic	
	11	9443192		2016-09-13	Cosic	

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS Remove

Examiner Initial*	Cite No	Publication Number	Kind Code[1]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	20030065662		2003-04-03	Cosic	
	2	20040194017		2004-09-30	Cosic	
	3	20050149517		2005-07-07	Cosic	
	4	20050149542		2005-07-07	Cosic	
	5	20050289105		2005-12-29	Cosic	
	6	20100023541		2010-01-28	Cosic	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	7	20100082536		2010-04-01	Cosic	
	8	20130218932		2013-08-22	Cosic	
	9	20130226974		2013-08-29	Cosic	
	10	20160140999		2016-05-19	Cosic	
	11	20160142650		2016-05-19	Cosic	
	12	20160246819		2016-08-25	Cosic	
	13	20160246850		2016-08-25	Cosic	
	14	20160246868		2016-08-25	Cosic	
	15	20160292185		2016-10-06	Cosic	

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number[3]	Country Code[2] i	Kind Code[4]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T[5]

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	1						

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T5
	1		

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature		Date Considered	

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Jasmin Cosic/	Date (YYYY-MM-DD)	2016-11-10
Name/Print	Jasmin Cosic	Registration Number	

This collection of information is required by 37 CFR 1.97 and 1.98. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 1 hour to complete, including gathering, preparing and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Electronic Acknowledgement Receipt

EFS ID:	27480807
Application Number:	15340991
International Application Number:	
Confirmation Number:	1993
Title of Invention:	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION
First Named Inventor/Applicant Name:	Jasmin Cosic
Customer Number:	116094
Filer:	Jasmin Cosic
Filer Authorized By:	
Attorney Docket Number:	
Receipt Date:	10-NOV-2016
Filing Date:	
Time Stamp:	20:16:32
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	no

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1	Information Disclosure Statement (IDS) Form (SB08)	IDS_Part1.pdf	615743 e8ef6a4647288aeeb4669422b73e856fbee955d6	no	13

Warnings:

Information:

2	Information Disclosure Statement (IDS) Form (SB08)	IDS_Part2.pdf	613126 c897c545b8c95a31e6ac9a660e4f9462410b4212	no	5

Warnings:

Information:

A U.S. Patent Number Citation or a U.S. Publication Number Citation is required in the Information Disclosure Statement (IDS) form for autoloading of data into USPTO systems. You may remove the form to add the required data in order to correct the Informational Message if you are citing U.S. References. If you chose not to include U.S. References, the image of the form will be processed and be made available within the Image File Wrapper (IFW) system. However, no data will be extracted from this form. Any additional data such as Foreign Patent Documents or Non Patent Literature will be manually reviewed and keyed into USPTO systems.

3	Information Disclosure Statement (IDS) Form (SB08)	IDS_Part3.pdf	614630 049e1bd89f595efb755d3fb7bfe296fc8a989a2b	no	10

Warnings:

Information:

4	Information Disclosure Statement (IDS) Form (SB08)	IDS_Part4.pdf	612917 e10b0c0937c6bb310686e40b525d3c11a97969f0	no	6

Warnings:

Information:

5	Non Patent Literature	NPL_Part1.pdf	19779115 ee07479784f30836d39a22bda6b5ec735694f56e	no	380

Warnings:

The page size in the PDF is too large. The pages should be 8.5 x 11 or A4. If this PDF is submitted, the pages will be resized upon entry into the Image File Wrapper and may affect subsequent processing

Information:

6	Non Patent Literature	NPL_Part2.pdf	5879608 8fd00edc18b82bf74a8ebf692fa5e39b231f3d56	no	114

Warnings:

The page size in the PDF is too large. The pages should be 8.5 x 11 or A4. If this PDF is submitted, the pages will be resized upon entry into the Image File Wrapper and may affect subsequent processing

Information:

7	Non Patent Literature	NPL_Part3.pdf	16903333 327e784f4ab311324ac99b3c6dd8b06ae0b170c7	no	160

Warnings:

Information:

Total Files Size (in bytes):	45018472

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.

National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

PATENT APPLICATION FEE DETERMINATION RECORD

Substitute for Form PTO-875

Application or Docket Number
15/340,991

APPLICATION AS FILED - PART I

FOR	NUMBER FILED (Column 1)	NUMBER EXTRA (Column 2)	SMALL ENTITY RATE($)	SMALL ENTITY FEE($)	OR	OTHER THAN SMALL ENTITY RATE($)	OTHER THAN SMALL ENTITY FEE($)
BASIC FEE (37 CFR 1.16(a), (b), or (c))	N/A	N/A	N/A	70		N/A	
SEARCH FEE (37 CFR 1.16(k), (i), or (m))	N/A	N/A	N/A	300		N/A	
EXAMINATION FEE (37 CFR 1.16(o), (p), or (q))	N/A	N/A	N/A	360		N/A	
TOTAL CLAIMS (37 CFR 1.16(i))	20 minus 20 =	*	x 40 =	0.00	OR		
INDEPENDENT CLAIMS (37 CFR 1.16(h))	3 minus 3 =	*	x 210 =	0.00			
APPLICATION SIZE FEE (37 CFR 1.16(s))	If the specification and drawings exceed 100 sheets of paper, the application size fee due is $310 ($155 for small entity) for each additional 50 sheets or fraction thereof. See 35 U.S.C. 41(a)(1)(G) and 37 CFR 1.16(s).			200			
MULTIPLE DEPENDENT CLAIM PRESENT (37 CFR 1.16(j))				0.00			
* If the difference in column 1 is less than zero, enter "0" in column 2.			TOTAL	930		TOTAL	

APPLICATION AS AMENDED - PART II

AMENDMENT A

	CLAIMS REMAINING AFTER AMENDMENT (Column 1)		HIGHEST NUMBER PREVIOUSLY PAID FOR (Column 2)	PRESENT EXTRA (Column 3)	SMALL ENTITY RATE($)	SMALL ENTITY ADDITIONAL FEE($)	OR	OTHER THAN SMALL ENTITY RATE($)	OTHER THAN SMALL ENTITY ADDITIONAL FEE($)
Total (37 CFR 1.16(i))	*	Minus	**	=	x =		OR	x =	
Independent (37 CFR 1.16(h))	*	Minus	***	=	x =		OR	x =	
Application Size Fee (37 CFR 1.16(s))									
FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))							OR		
					TOTAL ADD'L FEE		OR	TOTAL ADD'L FEE	

AMENDMENT B

	CLAIMS REMAINING AFTER AMENDMENT (Column 1)		HIGHEST NUMBER PREVIOUSLY PAID FOR (Column 2)	PRESENT EXTRA (Column 3)	RATE($)	ADDITIONAL FEE($)	OR	RATE($)	ADDITIONAL FEE($)
Total (37 CFR 1.16(i))	*	Minus	**	=	x =		OR	x =	
Independent (37 CFR 1.16(h))	*	Minus	***	=	x =		OR	x =	
Application Size Fee (37 CFR 1.16(s))									
FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))							OR		
					TOTAL ADD'L FEE		OR	TOTAL ADD'L FEE	

* If the entry in column 1 is less than the entry in column 2, write "0" in column 3.
** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, enter "20".
*** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, enter "3".
The "Highest Number Previously Paid For" (Total or Independent) is the highest found in the appropriate box in column 1.

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NUMBER	FILING or 371(c) DATE	GRP ART UNIT	FIL FEE REC'D	ATTY.DOCKET.NO	TOT CLAIMS	IND CLAIMS
15/340,991	11/02/2016		930		20	3

CONFIRMATION NO. 1993

FILING RECEIPT

OC000000087312368

116094
Jasmin Cosic
108 Woodbury Street
Pawtucket, RI 02861

Date Mailed: 11/21/2016

Receipt is acknowledged of this non-provisional patent application. The application will be taken up for examination in due course. Applicant will be notified as to the results of the examination. Any correspondence concerning the application must include the following identification information: the U.S. APPLICATION NUMBER, FILING DATE, NAME OF APPLICANT, and TITLE OF INVENTION. Fees transmitted by check or draft are subject to collection. Please verify the accuracy of the data presented on this receipt. **If an error is noted on this Filing Receipt, please submit a written request for a Filing Receipt Correction. Please provide a copy of this Filing Receipt with the changes noted thereon. If you received a "Notice to File Missing Parts" for this application, please submit any corrections to this Filing Receipt with your reply to the Notice. When the USPTO processes the reply to the Notice, the USPTO will generate another Filing Receipt incorporating the requested corrections**

Inventor(s)

Jasmin Cosic, Miami, FL;

Applicant(s)

Jasmin Cosic, Miami, FL;

Power of Attorney: None

Domestic Applications for which benefit is claimed - None.
A proper domestic benefit claim must be provided in an Application Data Sheet in order to constitute a claim for domestic benefit. See 37 CFR 1.76 and 1.78.

Foreign Applications for which priority is claimed (You may be eligible to benefit from the **Patent Prosecution Highway** program at the USPTO. Please see http://www.uspto.gov for more information.) - None.
Foreign application information must be provided in an Application Data Sheet in order to constitute a claim to foreign priority. See 37 CFR 1.55 and 1.76.

Permission to Access Application via Priority Document Exchange: No

Permission to Access Search Results: No

Applicant may provide or rescind an authorization for access using Form PTO/SB/39 or Form PTO/SB/69 as appropriate.

If Required, Foreign Filing License Granted: 11/17/2016

The country code and number of your priority application, to be used for filing abroad under the Paris Convention, is **US 15/340,991**

Projected Publication Date: Request for Non-Publication Acknowledged

Non-Publication Request: Yes

Early Publication Request: No
**** SMALL ENTITY ****
Title

ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION

Preliminary Class

Statement under 37 CFR 1.55 or 1.78 for AIA (First Inventor to File) Transition Applications: No

PROTECTING YOUR INVENTION OUTSIDE THE UNITED STATES

Since the rights granted by a U.S. patent extend only throughout the territory of the United States and have no effect in a foreign country, an inventor who wishes patent protection in another country must apply for a patent in a specific country or in regional patent offices. Applicants may wish to consider the filing of an international application under the Patent Cooperation Treaty (PCT). An international (PCT) application generally has the same effect as a regular national patent application in each PCT-member country. The PCT process **simplifies** the filing of patent applications on the same invention in member countries, but **does not result** in a grant of "an international patent" and does not eliminate the need of applicants to file additional documents and fees in countries where patent protection is desired.

Almost every country has its own patent law, and a person desiring a patent in a particular country must make an application for patent in that country in accordance with its particular laws. Since the laws of many countries differ in various respects from the patent law of the United States, applicants are advised to seek guidance from specific foreign countries to ensure that patent rights are not lost prematurely.

Applicants also are advised that in the case of inventions made in the United States, the Director of the USPTO must issue a license before applicants can apply for a patent in a foreign country. The filing of a U.S. patent application serves as a request for a foreign filing license. The application's filing receipt contains further information and guidance as to the status of applicant's license for foreign filing.

Applicants may wish to consult the USPTO booklet, "General Information Concerning Patents" (specifically, the section entitled "Treaties and Foreign Patents") for more information on timeframes and deadlines for filing foreign patent applications. The guide is available either by contacting the USPTO Contact Center at 800-786-9199, or it can be viewed on the USPTO website at http://www.uspto.gov/web/offices/pac/doc/general/index.html.

For information on preventing theft of your intellectual property (patents, trademarks and copyrights), you may wish to consult the U.S. Government website, http://www.stopfakes.gov. Part of a Department of Commerce initiative, this website includes self-help "toolkits" giving innovators guidance on how to protect intellectual property in specific countries such as China, Korea and Mexico. For questions regarding patent enforcement issues, applicants may call the U.S. Government hotline at 1-866-999-HALT (1-866-999-4258).

LICENSE FOR FOREIGN FILING UNDER

Title 35, United States Code, Section 184

Title 37, Code of Federal Regulations, 5.11 & 5.15

GRANTED

The applicant has been granted a license under 35 U.S.C. 184, if the phrase "IF REQUIRED, FOREIGN FILING LICENSE GRANTED" followed by a date appears on this form. Such licenses are issued in all applications where the conditions for issuance of a license have been met, regardless of whether or not a license may be required as set forth in 37 CFR 5.15. The scope and limitations of this license are set forth in 37 CFR 5.15(a) unless an earlier license has been issued under 37 CFR 5.15(b). The license is subject to revocation upon written notification. The date indicated is the effective date of the license, unless an earlier license of similar scope has been granted under 37 CFR 5.13 or 5.14.

This license is to be retained by the licensee and may be used at any time on or after the effective date thereof unless it is revoked. This license is automatically transferred to any related applications(s) filed under 37 CFR 1.53(d). This license is not retroactive.

The grant of a license does not in any way lessen the responsibility of a licensee for the security of the subject matter as imposed by any Government contract or the provisions of existing laws relating to espionage and the national security or the export of technical data. Licensees should apprise themselves of current regulations especially with respect to certain countries, of other agencies, particularly the Office of Defense Trade Controls, Department of State (with respect to Arms, Munitions and Implements of War (22 CFR 121-128)); the Bureau of Industry and Security, Department of Commerce (15 CFR parts 730-774); the Office of Foreign AssetsControl, Department of Treasury (31 CFR Parts 500+) and the Department of Energy.

NOT GRANTED

No license under 35 U.S.C. 184 has been granted at this time, if the phrase "IF REQUIRED, FOREIGN FILING LICENSE GRANTED" DOES NOT appear on this form. Applicant may still petition for a license under 37 CFR 5.12, if a license is desired before the expiration of 6 months from the filing date of the application. If 6 months has lapsed from the filing date of this application and the licensee has not received any indication of a secrecy order under 35 U.S.C. 181, the licensee may foreign file the application pursuant to 37 CFR 5.15(b).

PTO/SB/06 (09-11)
Approved for use through 1/31/2014. OMB 0651-0032
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

PATENT APPLICATION FEE DETERMINATION RECORD Substitute for Form PTO-875	Application or Docket Number 15/340,991	Filing Date 11/02/2016	☐ To be Mailed

ENTITY: ☐ LARGE ☑ SMALL ☐ MICRO

APPLICATION AS FILED - PART I

FOR	(Column 1) NUMBER FILED	(Column 2) NUMBER EXTRA	RATE ($)	FEE ($)
☐ BASIC FEE (37 CFR 1.16(a), (b), or (c))	N/A	N/A	N/A	
☐ SEARCH FEE (37 CFR 1.16(k), (i), or (m))	N/A	N/A	N/A	
☐ EXAMINATION FEE (37 CFR 1.16(o), (p), or (q))	N/A	N/A	N/A	
TOTAL CLAIMS (37 CFR 1.16(i))	minus 20 =	*	x $40 =	
INDEPENDENT CLAIMS (37 CFR 1.16(h))	minus 3 =	*	x $210 =	
☐ APPLICATION SIZE FEE (37 CFR 1.16(s))	If the specification and drawings exceed 100 sheets of paper, the application size fee due is $310 ($155 for small entity) for each additional 50 sheets or fraction thereof. See 35 U.S.C. 41(a)(1)(G) and 37 CFR 1.16(s).			
☐ MULTIPLE DEPENDENT CLAIM PRESENT (37 CFR 1.16(j))				
* If the difference in column 1 is less than zero, enter "0" in column 2.			TOTAL	

APPLICATION AS AMENDED - PART II

AMENDMENT	05/04/2019	(Column 1) CLAIMS REMAINING AFTER AMENDMENT		(Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR	(Column 3) PRESENT EXTRA	RATE ($)	ADDITIONAL FEE ($)
	Total (37 CFR 1.16(i))	* 20	Minus	** 20	= 0	x $50 =	0
	Independent (37 CFR 1.16(h))	* 3	Minus	*** 3	= 0	x $230 =	0
	☐ Application Size Fee (37 CFR 1.16(s))						
	☐ FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))						
						TOTAL ADD'L FEE	0

AMENDMENT		(Column 1) CLAIMS REMAINING AFTER AMENDMENT		(Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR	(Column 3) PRESENT EXTRA	RATE ($)	ADDITIONAL FEE ($)
	Total (37 CFR 1.16(i))	*	Minus	**	=	x $0 =	
	Independent (37 CFR 1.16(h))	*	Minus	***	=	x $0 =	
	☐ Application Size Fee (37 CFR 1.16(s))						
	☐ FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))						
						TOTAL ADD'L FEE	

* If the entry in column 1 is less than the entry in column 2, write "0" in column 3.	LIE
** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, enter "20".	/WILLIAM N PHILLIPS/
*** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, enter "3".	
The "Highest Number Previously Paid For" (Total or Independent) is the highest number found in the appropriate box in column 1.	

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Inventor:	Jasmin Cosic	Confirmation No.: 1993
Title:	"ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION"	
Serial No.:	15/340,991	Filed: November 2, 2016
Examiner:	PARK, SOO JIN	Group Art Unit: 2668

Via EFS-Web May 4, 2019

Mail Stop Amendment
COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, VA 22313-1450

PRELIMINARY AMENDMENT

Dear Commissioner:

Before examination on the merits, please amend the claims in the above-identified application as shown in the following listing of claims.

Electronic Acknowledgement Receipt

EFS ID:	35919394
Application Number:	15340991
International Application Number:	
Confirmation Number:	1993
Title of Invention:	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION
First Named Inventor/Applicant Name:	Jasmin Cosic
Customer Number:	116094
Filer:	Jasmin Cosic
Filer Authorized By:	
Attorney Docket Number:	
Receipt Date:	04-MAY-2019
Filing Date:	02-NOV-2016
Time Stamp:	22:35:52
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	no

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1		PRELIMINARY_AMENDMENT.pdf	40926 6bf4f7439b644297b38609aca6b1923e32fa2f87	yes	11

Multipart Description/PDF files in .zip description		
Document Description	**Start**	**End**
Applicant Arguments/Remarks Made in an Amendment	11	11
Claims	2	10
Preliminary Amendment	1	1

Warnings:

Information:

Total Files Size (in bytes):	40926

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.
National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.
New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

Inventor: Jasmin Cosic
Serial No.: 15/340,991
Filing Date: November 2, 2016

Remarks

In this preliminary amendment, the applicant cancels original claims 1-20 and presents for examination new claims 21-40. After entry of this preliminary amendment, claims 21-40 are pending. The undersigned may be contacted at (317) 772-1312 concerning this application.

I hereby certify that this correspondence is being submitted electronically via EFS Web to the United States Patent and Trademark Office.

By /Jasmin Cosic/
Jasmin Cosic

Date submitted: May 4, 2019

Respectfully submitted,

/Jasmin Cosic/

Jasmin Cosic

Listing of Claims

1 - 20 (Canceled)

21. (new) A system comprising:

one or more processor circuits;

a memory that stores at least a first correlation including a first one or more object representations correlated with a first one or more instruction sets for operating a first device, wherein the first device includes a first one or more sensors, and wherein the first one or more object representations represent a first one or more objects detected at least in part by the first one or more sensors; and

an artificial intelligence unit that:

generates or receives a second one or more object representations, wherein the second one or more object representations represent a second one or more objects;

determines the first one or more instruction sets for operating the first device based on at least partial match between the second one or more object representations and the first one or more object representations; and

in response to the determines of the artificial intelligence unit, causes the first device or a second device to perform one or more operations defined by the first one or more instruction sets for operating the first device at least by causing

the one or more processor circuits to execute the first one or more instruction sets for operating the first device.

22. (new) The system of claim 21, wherein at least one object of the first one or more objects and at least one object of the second one or more objects are the same.

23. (new) The system of claim 21, wherein the determines the first one or more instruction sets for operating the first device based on the at least partial match between the second one or more object representations and the first one or more object representations includes:

determining that a number of at least partially matching portions of the second one or more object representations and portions of the first one or more object representations exceeds a threshold number; or

determining that a percentage of at least partially matching portions of the second one or more object representations and portions of the first one or more object representations exceeds a threshold percentage.

24. (new) The system of claim 21, wherein the memory further stores at least a first knowledge cell, and wherein the first knowledge cell includes the first correlation.

25. (new) The system of claim 21, wherein the first one or more object representations include a first stream of one or more object representations, and wherein the second one or more object representations include a second stream of one or more object representations.

26. (new) The system of claim 21, wherein, to correlate the first one or more object representations with the first one or more instruction sets for operating the first device, a determination is made that the first one or more instruction sets for operating the first device temporally correspond to the first one or more object representations.

27. (new) The system of claim 21, wherein the artificial intelligence unit includes at least one selected from the group consisting of: a hardware element that is included in the one or more processor circuits, a hardware element that is included in another one or more processor circuits, a program operating on the one or more processor circuits, a program operating on another one or more processor circuits, and an element coupled to the one or more processor circuits, and wherein the first device includes: a robot, a vehicle, an appliance, an electronic device, or a mechanical machine, and wherein the second device includes: a robot, a vehicle, an appliance, an electronic device, or a mechanical machine.

28. (new) The system of claim 21, wherein the memory further stores at least a third correlation including a third one or more object representations correlated with a third one or more instruction sets for operating the first device, and wherein the third one or more object representations represent a third one or more objects detected at least in part by the first one or more sensors, and wherein a first connection is generated to connect the first correlation with the third correlation, and wherein the first correlation connected with the third correlation form at least a portion of a knowledge structure or a knowledgebase.

29. (new) The system of claim 21, wherein the memory further stores at least a third correlation including a third one or more object representations correlated with a third one or more instruction sets for operating the second device, and wherein the second device includes a second one or more sensors, and wherein the third one or more object representations represent a third one or more objects detected at least in part by the second one or more sensors.

30. (new) The system of claim 21, wherein at least a portion of the first correlation is learned in a learning process while the first device is at least partially operated by a user, and wherein the learning process includes:

generating or receiving the first one or more object representations; and

obtaining or receiving the first one or more instruction sets for operating the first device.

31. (new) The system of claim 21, wherein the artificial intelligence unit includes at least one selected from the group consisting of: at least a portion of an object processing unit, at least a portion of an acquisition interface, at least a portion of a modification interface, and at least a portion of a DCADO unit.

32. (new) The system of claim 21, wherein the second one or more objects include one or more objects detected at least in part by the first one or more sensors, and wherein the first one or more instruction sets for operating the first device are applied to the first device so that the first device performs the one or more operations defined by the first one or more instruction sets for operating the first device.

33. (new) The system of claim 21, wherein the second device includes a second one or more sensors, and wherein the second one or more objects include one or more objects detected at least in part by the second one or more sensors, and wherein the first one or more instruction sets for operating the first device are applied to the second device so that the second device performs the one or more operations defined by the first one or more instruction sets for operating the first device.

34. (new) The system of claim 21, wherein the second device includes a second one or more sensors, and wherein the second one or more objects include one or more objects detected at least in part by the second one or more sensors, and

wherein the first one or more instruction sets for operating the first device are modified and applied to the second device so that the second device performs the one or more operations defined by the modified first one or more instruction sets for operating the first device.

35. (new) A non-transitory machine readable medium having a program stored thereon that when executed by one or more processor circuits causes the one or more processor circuits to perform operations comprising:

accessing a memory that stores at least a first correlation including a first one or more object representations correlated with a first one or more instruction sets for operating a first device, wherein the first device includes a first one or more sensors, and wherein the first one or more object representations represent a first one or more objects detected at least in part by the first one or more sensors;

generating or receiving a second one or more object representations, wherein the second one or more object representations represent a second one or more objects;

determining the first one or more instruction sets for operating the first device based on at least partial match between the second one or more object representations and the first one or more object representations; and

in response to the determining, causing the first device or a second device to perform one or more operations defined by the first one or more instruction sets for operating the first device at least by causing the one or more processor

circuits or another one or more processor circuits to execute the first one or more instruction sets for operating the first device.

36. (new) The non-transitory machine readable medium of claim 35, wherein, to correlate the first one or more object representations with the first one or more instruction sets for operating the first device, a determination is made that the first one or more instruction sets for operating the first device temporally correspond to the first one or more object representations.

37. (new) The non-transitory machine readable medium of claim 35, wherein the second device includes a second one or more sensors, and wherein the second one or more objects include one or more objects detected at least in part by the second one or more sensors, and wherein the first one or more instruction sets for operating the first device are applied to the second device so that the second device performs the one or more operations defined by the first one or more instruction sets for operating the first device.

38. (new) A method comprising:

(a) accessing a memory that stores at least a first correlation including a first one or more object representations correlated with a first one or more instruction sets for operating a first device, wherein the first device includes a first one or more sensors, and wherein the first one or more object representations

represent a first one or more objects detected at least in part by the first one or more sensors, the accessing of (a) performed by one or more processor circuits;

(b) generating or receiving a second one or more object representations, wherein the second one or more object representations represent a second one or more objects, the generating or the receiving of (b) performed by the one or more processor circuits;

(c) determining the first one or more instruction sets for operating the first device based on at least partial match between the second one or more object representations and the first one or more object representations, the determining of (c) performed by the one or more processor circuits;

(d) executing the first one or more instruction sets for operating the first device, the executing of (d) performed by the one or more processor circuits or by another one or more processor circuits in response to the determining of (c); and

(e) performing, by the first device or by a second device, one or more operations defined by the first one or more instruction sets for operating the first device.

39. (new) The method of claim 38, wherein, to correlate the first one or more object representations with the first one or more instruction sets for operating the first device, a determination is made that the first one or more instruction sets for operating the first device temporally correspond to the first one or more object representations.

40. (new) The method of claim 38, wherein the second device includes a second one or more sensors, and wherein the second one or more objects include one or more objects detected at least in part by the second one or more sensors, and wherein the first one or more instruction sets for operating the first device are applied to the second device so that the second device performs the one or more operations defined by the first one or more instruction sets for operating the first device.

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

PTO/SB/08a (01-10)
Approved for use through 07/31/2012. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS Remove

Examiner Initial*	Cite No	Patent Number	Kind Code[1]	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	8996432		2015-03-31	Fu; Jicheng	
	2	6754631		2004-06-22	Din	
	3	9305216		2016-04-05	Mishra	
	4	5560011		1996-09-23	Uyama	
	5	6842877		2005-01-11	Robarts , et al.	
	6	7565340		2009-07-21	Herlocker , et al.	
	7	8261199		2012-09-04	Cradick , et al.	
	8	8266608		2012-09-11	Hecht , et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	9	9268454		2016-02-23	Hamilton, II , et al.	

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS

Remove

Examiner Initial*	Cite No	Publication Number	Kind Code[1]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	20120290347		2012-11-15	ELAZOUNI; ASHRAF ; et al.	
	2	20150324685		2015-11-12	Bohn; Richard Esten ; et al.	
	3	20100114746		2010-05-06	Bobbitt; Russell Patrick ; et al.	
	4	20120284026		2012-11-08	Cardillo; Peter S. ; et al.	
	5	20140211988		2014-07-31	Fan; Quanfu ; et al.	
	6	20150006171		2015-01-01	Westby; Michael C. ; et al.	
	7	20150264306		2015-09-17	Marilly; Emmanuel ; et al.	
	8	20110085734		2011-04-14	Berg; Jared S. ; et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	9	20150269415		2015-09-24	Gelbman; Alexander	
	10	20160274187		2016-09-22	MENON; SANKARAN M. ; et al.	
	11	20150339213		2015-11-22	LEE; Christopher Stephen ; et al.	
	12	20150310041		2015-10-29	Kier; Scott ; et al.	
	13	20160328480		2016-11-10	Owens; Erich James ; et al.	
	14	20140075249		2014-03-13	SATO; Shuhei ; et al.	
	15	20110270794		2011-11-03	Drory; Tal ; et al.	
	16	20070050719		2007-03-01	Lui; Philip ; et al.	
	17	20070061735		2007-03-15	Hoffberg; Steven M. ; et al.	
	18	20060184410		2006-08-17	Ramamurthy; Shankar ; et al.	
	19	20060265406		2006-11-23	Chkodrov; Gueorgui B. ; et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	20	20090287643		2009-11-19	Corville; Allen O. ; et al.	
	21	20110007079		2011-01-13	Perez et al.	
	22	20070050606		2007-03-01	Ferren et al.	
	23	20090067727		2009-03-12	HIROHATA; Hitoshi	
	24	20130007532		2013-01-03	Miller et al.	
	25	20050240412		2005-10-27	Fujita	
	26	20160167226		2016-06-16	Schnittman	
	27	20090136095		2009-05-28	Marcon et al.	
	28	20080215508		2008-09-04	Hanneman; Jeffrey E. ; et al.	
	29	20090044113		2009-02-12	Jones; Scott T. ; et al.	
	30	20130278501		2013-10-24	BULZACKI; Adrian	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number[3]	Country Code[2] i	Kind Code[4]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T[5]
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T[5]
	1	Chen et al. "Case-Based Reasoning System and Artificial Neural Networks: A Review Neural Comput & Applic (2001) 10: pp 264-276, 13 pages	
	2	JOHN J. GREFENSTETTE, CONNIE LOGGIA RAMSEY, ALAN C. SCHULTZ, Learning Sequential Decision Rules Using Simulation Models and Competition, 1990, Navy Center for Applied Research in Artificial Intelligence, Naval Research Laboratory, Washington, DC, 27 pages	
	3	ALAN C. SCHULTZ, JOHN J. GREFENSTETTE, Using a Genetic Algorithm to Learn Behaviors for Autonomous Vehicles, 1992, Navy Center for Applied Research in Artificial Intelligence, Naval Research Laboratory, Washington, DC, 12 pages	
	4	Koppula et al., "Anticipating human activities using object affordances for reactive robotic response", IEEE TRAMI 2016, published 5 May 2015, 16 pages	
	5	Orme, "System design tips for entry level smartphones - part 3", found online at "https://community.arm.com/processors/b/blog/posts/system-design-tips-for-entry-level-smartphones---part-3", Oct 21, 2013, 12 pages	
	6	NOWOSTAWSKI, MARIUSZ et al., "Dynamic Demes Parallel Genetic Algorithm", May 13, 1999, 6 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	7	Stack Overflow, How can I quantify difference between two images?, accessed 2 August 2017 at https://stackoverflow.com/questions/189943/how-can-i-quantify-difference-between-two-images, 8 pages	
	8	SKOLICKI, ZBIGNIEW et al., "The Influence of Migration Sizes and Intervals on Island Models", June 29, 2005, 8 pages	
	9	Vahid Lari, et al., "Decentralized dynamic resource management support for massively parallel processor arrays", September 11, 2011, 8 pages	
	10	Vahid Lari, et al., "Distributed resource reservation in massively parallel processor arrays", May 16, 2011, 4 pages	
	11	Mohsen Hayati and Yazdan Shirvany, "Artificial Neural Network Approach for Short Term Load Forecasting for Illam Region", January 2007, 5 pages	
	12	Hasim Sak, Andrew Senior, and Francoise Beaufays, "Long Short-Term Memory Recurrent Neural Network Architectures for Large Scale Acoustic Modeling", January 2014, 5 pages	
	13	Jorg Walter and Klaus Schulten, "Implementation of self-organizing neural networks for visuo-motor control of an industrial robot", January 1993, 10 pages	
	14	Heikki Hyotyniemi and Aarno Lehtola, "A Universal Relation Database Interface for Knowledge Based Systems", April 1991, 5 pages	
	15	Mrissa, Michael, et al. "An avatar architecture for the web of things." IEEE Internet Computing 19.2 (2015): 30-38., 9 pages	
	16	Luck, Michael, and Ruth Aylett. "Applying artificial intelligence to virtual reality: Intelligent virtual environments." Applied Artificial Intelligence 14.1 (2000): 3-32., 30 pages	
	17	Terdjimi, Mehdi, et al. "An avatar-based adaptation workflow for the web of things." Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), 2016 IEEE 25th International Conference on. IEEE, 2016., 6 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	18	Bogdanovych, Anton, et al. "Authentic interactive reenactment of cultural heritage with 3D virtual worlds and artificial intelligence." Applied Artificial Intelligence 24.6 (2010): 617-647., 32 pages	
	19	Hernandez, Marco E. Perez, and Stephan Reiff-Marganiec. "Autonomous and self controlling smart objects for the future internet." Future internet of things and cloud (FiCloud), 2015 3rd international conference on. IEEE, 2015., 8 pages	
	20	Medini et al., "Building a Web of Things with Avatars", Managing the Web of Things (2017), 30 pages	

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature		Date Considered	

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Jasmin Cosic/	Date (YYYY-MM-DD)	2019-05-05
Name/Print	Jasmin Cosic	Registration Number	

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Electronic Acknowledgement Receipt

EFS ID:	35919853
Application Number:	15340991
International Application Number:	
Confirmation Number:	1993
Title of Invention:	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION
First Named Inventor/Applicant Name:	Jasmin Cosic
Customer Number:	116094
Filer:	Jasmin Cosic
Filer Authorized By:	
Attorney Docket Number:	
Receipt Date:	05-MAY-2019
Filing Date:	02-NOV-2016
Time Stamp:	15:18:25
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	no

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1	Information Disclosure Statement (IDS) Form (SB08)	IDS_Part5.pdf	619072 872308068d078ac226845930811367d9be024dd2	no	9

Warnings:

Information:

2	Non Patent Literature	NPL_Part5.pdf	25533925 58bed29843cae9904ec69a4efe47cc83b3ece30a	no	254

Warnings:

The page size in the PDF is too large. The pages should be 8.5 x 11 or A4. If this PDF is submitted, the pages will be resized upon entry into the Image File Wrapper and may affect subsequent processing

Information:

Total Files Size (in bytes):	26152997

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.
National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.
New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

PTO/SB/06 (09-11)
Approved for use through 1/31/2014. OMB 0651-0032
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

PATENT APPLICATION FEE DETERMINATION RECORD Substitute for Form PTO-875	Application or Docket Number 15/340,991	Filing Date 11/02/2016	☐ To be Mailed

ENTITY: ☐ LARGE ☑ SMALL ☐ MICRO

APPLICATION AS FILED - PART I

FOR	(Column 1) NUMBER FILED	(Column 2) NUMBER EXTRA		RATE ($)	FEE ($)
☐ BASIC FEE (37 CFR 1.16(a), (b), or (c))	N/A	N/A		N/A	
☐ SEARCH FEE (37 CFR 1.16(k), (i), or (m))	N/A	N/A		N/A	
☐ EXAMINATION FEE (37 CFR 1.16(o), (p), or (q))	N/A	N/A		N/A	
TOTAL CLAIMS (37 CFR 1.16(i))	minus 20 =	*		x $40 =	
INDEPENDENT CLAIMS (37 CFR 1.16(h))	minus 3 =	*		x $210 =	
☐ APPLICATION SIZE FEE (37 CFR 1.16(s))	If the specification and drawings exceed 100 sheets of paper, the application size fee due is $310 ($155 for small entity) for each additional 50 sheets or fraction thereof. See 35 U.S.C. 41(a)(1)(G) and 37 CFR 1.16(s).				
☐ MULTIPLE DEPENDENT CLAIM PRESENT (37 CFR 1.16(j))					
* If the difference in column 1 is less than zero, enter "0" in column 2.				TOTAL	

APPLICATION AS AMENDED - PART II

AMENDMENT	05/27/2019	(Column 1) CLAIMS REMAINING AFTER AMENDMENT		(Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR	(Column 3) PRESENT EXTRA		RATE ($)	ADDITIONAL FEE ($)
	Total (37 CFR 1.16(i))	* 20	Minus	** 20	= 0		x $50 =	0
	Independent (37 CFR 1.16(h))	* 3	Minus	*** 3	= 0		x $230 =	0
	☐ Application Size Fee (37 CFR 1.16(s))							
	☐ FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))							
							TOTAL ADD'L FEE	0

AMENDMENT		(Column 1) CLAIMS REMAINING AFTER AMENDMENT		(Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR	(Column 3) PRESENT EXTRA		RATE ($)	ADDITIONAL FEE ($)
	Total (37 CFR 1.16(i))	*	Minus	**	=		x $0 =	
	Independent (37 CFR 1.16(h))	*	Minus	***	=		x $0 =	
	☐ Application Size Fee (37 CFR 1.16(s))							
	☐ FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))							
							TOTAL ADD'L FEE	

* If the entry in column 1 is less than the entry in column 2, write "0" in column 3.	LIE
** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, enter "20".	/ANGELA S WHITE/
*** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, enter "3".	
The "Highest Number Previously Paid For" (Total or Independent) is the highest number found in the appropriate box in column 1.	

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Inventor: Jasmin Cosic Confirmation No.: 1993

Title: "ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION"

Serial No.: 15/340,991 Filed: November 2, 2016

Examiner: PARK, SOO JIN Group Art Unit: 2668

Via EFS-Web May 27, 2019

Mail Stop Amendment
COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, VA 22313-1450

PRELIMINARY AMENDMENT

Dear Commissioner:

Before examination on the merits, please amend the claims in the above-identified application as shown in the following listing of claims. This preliminary amendment supersedes the preliminary amendment filed on May 4, 2019, therefore, the claims in the following listing of claims should be used for the examination on the merits instead of the claims in the preliminary amendment filed on May 4, 2019.

Electronic Acknowledgement Receipt

EFS ID:	36119055
Application Number:	15340991
International Application Number:	
Confirmation Number:	1993
Title of Invention:	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION
First Named Inventor/Applicant Name:	Jasmin Cosic
Customer Number:	116094
Filer:	Jasmin Cosic
Filer Authorized By:	
Attorney Docket Number:	
Receipt Date:	27-MAY-2019
Filing Date:	02-NOV-2016
Time Stamp:	19:58:52
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	no

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1		PRELIMINARY_AMENDMENT.pdf	47718 9af1a7f9b33fbe1848dd1e377d965a60824bd8f6	yes	13

Multipart Description/PDF files in .zip description

Document Description	Start	End
Applicant Arguments/Remarks Made in an Amendment	13	13
Claims	2	12
Preliminary Amendment	1	1

Warnings:

Information:

Total Files Size (in bytes):	47718

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.
National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.
New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

Inventor: Jasmin Cosic
Serial No.: 15/340,991
Filing Date: November 2, 2016

Remarks

In this preliminary amendment, the applicant cancels claims 1-40, and presents for examination new claims 41-60. After entry of this preliminary amendment, claims 41-60 are pending. The undersigned may be contacted at (317) 772-1312 concerning this application.

I hereby certify that this correspondence is being submitted electronically via EFS Web to the United States Patent and Trademark Office.

By /Jasmin Cosic/
Jasmin Cosic

Date submitted: May 27, 2019

Respectfully submitted,

/Jasmin Cosic/

Jasmin Cosic

Inventor: Jasmin Cosic
Serial No.: 15/340,991
Filing Date: November 2, 2016

Listing of Claims

1 - 40 (Canceled)

41. (new) A method implemented using a computing system that includes one or more processor circuits, the method comprising:

accessing a memory that stores at least a first correlation including a first one or more object representations correlated with a first one or more instruction sets for operating a first device, wherein the first device includes a first one or more sensors, and wherein the first one or more object representations represent a first one or more objects detected at least in part by the first one or more sensors;

generating or receiving a second one or more object representations, wherein the second one or more object representations represent a second one or more objects;

determining the first one or more instruction sets for operating the first device based on at least partial match between the second one or more object representations and the first one or more object representations; and

at least in response to the determining, executing the first one or more instruction sets for operating the first device, wherein the first device or a second device performs one or more operations defined by the first one or more instruction sets for operating the first device.

42. (new) The method of claim 41, wherein at least one object of the first one or more objects and at least one object of the second one or more objects are the same.

43. (new) The method of claim 41, wherein the determining the first one or more instruction sets for operating the first device based on the at least partial match between the second one or more object representations and the first one or more object representations includes:

determining that a number of at least partially matching portions of the second one or more object representations and portions of the first one or more object representations exceeds a threshold number; or

determining that a percentage of at least partially matching portions of the second one or more object representations and portions of the first one or more object representations exceeds a threshold percentage.

44. (new) The method of claim 41, wherein the memory further stores at least a first knowledge cell, and wherein the first knowledge cell includes the first correlation.

45. (new) The method of claim 41, wherein the first one or more object representations include a first stream of one or more object representations, and

wherein the second one or more object representations include a second stream of one or more object representations.

46. (new) The method of claim 41, wherein, to correlate the first one or more object representations with the first one or more instruction sets for operating the first device, a determination is made that the first one or more instruction sets for operating the first device temporally correspond to the first one or more object representations.

47. (new) The method of claim 41, wherein elements of the computing system are included in: a single device or multiple devices, and wherein the one or more processor circuits include: one or more microcontrollers, one or more computing circuits, or one or more electronic circuits, and wherein the memory includes: a volatile memory or a non-volatile memory, and wherein the first device includes: a robot, a vehicle, an appliance, an electronic device, or a mechanical machine, and wherein the second device includes: a robot, a vehicle, an appliance, an electronic device, or a mechanical machine, and wherein the first one or more object representations include: one or more object properties of the first one or more objects, or one or more information on the first one or more objects, and wherein the second one or more object representations include: one or more object properties of the second one or more objects, or one or more information on the second one or more objects, and wherein an instruction set of the first one or more instruction sets for operating the first device includes at least one

selected from the group comprising: only one instruction, a plurality of instructions, one or more inputs, one or more commands, one or more computer commands, one or more keywords, one or more symbols, one or more operators, one or more variables, one or more values, one or more objects or object references, one or more data structures or data structure references, one or more functions or function references, one or more parameters, one or more signals, one or more characters, one or more digits, one or more user operating directions, one or more user directions, one or more user inputs, one or more representations of one or more user actions, one or more representations of one or more user clicks, one or more data, and one or more information, and wherein the first one or more sensors include at least one selected from the group comprising: one or more cameras, one or more microphones, one or more radars, one or more lidars, one or more sonars, and one or more apparatuses for detecting objects or object properties.

48. (new) The method of claim 41, wherein the memory further stores at least a third correlation including a third one or more object representations correlated with a third one or more instruction sets for operating the first device, and wherein the third one or more object representations represent a third one or more objects detected at least in part by the first one or more sensors, and wherein a first connection is generated to connect the first correlation with the third correlation, and wherein the first correlation connected with the third correlation form at least a portion of a knowledge structure or a knowledgebase

for operating at least one selected from the group comprising: the first device, the second device, and other one or more devices.

49. (new) The method of claim 41, wherein at least a portion of the first correlation is learned in a first learning process that includes:

generating or receiving the first one or more object representations; and

obtaining or receiving the first one or more instruction sets for operating the first device, and wherein the memory further stores at least a third correlation including a third one or more object representations correlated with a third one or more instruction sets for operating the second device, and wherein the second device includes a second one or more sensors, and wherein the third one or more object representations represent a third one or more objects detected at least in part by the second one or more sensors, and wherein at least a portion of the third correlation is learned in a second learning process that includes:

generating or receiving the third one or more object representations; and

obtaining or receiving the third one or more instruction sets for operating the second device.

50. (new) The method of claim 41, wherein the second one or more objects include one or more objects detected at least in part by: the first one or more sensors, or one or more sensors of the second device.

51. (new) The method of claim 41, further comprising:

modifying the first one or more instruction sets for operating the first device or a copy of the first one or more instruction sets for operating the first device, and wherein the executing the first one or more instruction sets for operating the first device includes executing the modified the first one or more instruction sets for operating the first device or the modified the copy of the first one or more instruction sets for operating the first device, and wherein the performing, by the first device or by the second device, the one or more operations defined by the first one or more instruction sets for operating the first device includes performing, by the first device or by the second device, the one or more operations defined by the modified the first one or more instruction sets for operating the first device or by the modified the copy of the first one or more instruction sets for operating the first device.

52. (new) The method of claim 41, wherein the first one or more object representations include or are associated with a first one or more extra information, and wherein the second one or more object representations include or are associated with a second one or more extra information, and wherein the determining the first one or more instruction sets for operating the first device based on the at least partial match between the second one or more object representations and the first one or more object representations is further based on at least partial match between the second one or more extra information and the first one or more extra information.

53. (new) The method of claim 41, wherein the memory further stores at least a third correlation including a third one or more object representations correlated with a third one or more instruction sets for operating the first device, and wherein the third one or more object representations represent a third one or more objects detected at least in part by the first one or more sensors, and wherein at least a portion of the first correlation is learned in a first learning process that includes operating the first device at least partially by a first user, and wherein at least a portion of the third correlation is learned in a second learning process that includes operating the first device at least partially by the first user.

54. (new) The method of claim 41, wherein the memory further stores at least a third correlation including a third one or more object representations correlated with a third one or more instruction sets for operating the first device, and wherein the third one or more object representations represent a third one or more objects detected at least in part by the first one or more sensors, and wherein at least a portion of the first correlation is learned in a first learning process that includes operating the first device at least partially by a first user, and wherein at least a portion of the third correlation is learned in a second learning process that includes operating the first device at least partially by a second user.

55. (new) The method of claim 41, wherein the memory further stores at least a third correlation including a third one or more object representations correlated with a third one or more instruction sets for operating the second device, and wherein the second device includes a second one or more sensors, and wherein the third one or more object representations represent a third one or more objects detected at least in part by the second one or more sensors, and wherein at least a portion of the first correlation is learned in a first learning process that includes operating the first device at least partially by a first user, and wherein at least a portion of the third correlation is learned in a second learning process that includes operating the second device at least partially by a second user.

56. (new) One or more non-transitory machine readable media storing machine readable code that when executed by one or more processor circuits causes the one or more processor circuits to perform at least:

accessing a memory that stores at least a first correlation including a first one or more object representations correlated with a first one or more instruction sets for operating a first device, wherein the first device includes a first one or more sensors, and wherein the first one or more object representations represent a first one or more objects detected at least in part by the first one or more sensors;

generating or receiving a second one or more object representations, wherein the second one or more object representations represent a second one or more objects;

determining the first one or more instruction sets for operating the first device based on at least partial match between the second one or more object representations and the first one or more object representations; and

at least in response to the determining, causing the first device or a second device to perform one or more operations defined by the first one or more instruction sets for operating the first device at least by causing the one or more processor circuits or another one or more processor circuits to execute the first one or more instruction sets for operating the first device.

57. (new) The one or more non-transitory machine readable media of claim 56, wherein, to correlate the first one or more object representations with the first one or more instruction sets for operating the first device, a determination is made that the first one or more instruction sets for operating the first device temporally correspond to the first one or more object representations.

58. (new) The one or more non-transitory machine readable media of claim 56, wherein the memory further stores at least a third correlation including a third one or more object representations correlated with a third one or more instruction sets for operating the first device, and wherein the third one or more object representations represent a third one or more objects detected at least in part by the first one or more sensors, and wherein at least a portion of the first correlation is learned in a first learning process that includes operating the first device at least partially by a first user, and wherein at least a portion of the third

correlation is learned in a second learning process that includes operating the first device at least partially by the first user.

59. (new) A system comprising:

a memory that stores at least a first correlation including a first one or more object representations correlated with a first one or more instruction sets for operating a first device, wherein the first device includes a first one or more sensors, and wherein the first one or more object representations represent a first one or more objects detected at least in part by the first one or more sensors;

a means for generating or receiving a second one or more object representations, wherein the second one or more object representations represent a second one or more objects;

a means for determining the first one or more instruction sets for operating the first device based on at least partial match between the second one or more object representations and the first one or more object representations; and

a means for executing the first one or more instruction sets for operating the first device at least in response to the determining, wherein the first device or a second device performs one or more operations defined by the first one or more instruction sets for operating the first device.

60. (new) The system of claim 59, wherein the means for generating or receiving the second one or more object representations includes one or more

processor circuits, and wherein the means for determining the first one or more instruction sets for operating the first device based on the at least partial match between the second one or more object representations and the first one or more object representations includes one or more processor circuits, and wherein the means for executing the first one or more instruction sets for operating the first device includes one or more processor circuits.

PTO/SB/06 (09-11)
Approved for use through 1/31/2014. OMB 0651-0032
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

PATENT APPLICATION FEE DETERMINATION RECORD Substitute for Form PTO-875	Application or Docket Number 15/340,991	Filing Date 11/02/2016	☐ To be Mailed

ENTITY: ☐ LARGE ☑ SMALL ☐ MICRO

APPLICATION AS FILED - PART I

	(Column 1)	(Column 2)			
FOR	NUMBER FILED	NUMBER EXTRA		RATE ($)	FEE ($)
☐ BASIC FEE (37 CFR 1.16(a), (b), or (c))	N/A	N/A		N/A	
☐ SEARCH FEE (37 CFR 1.16(k), (i), or (m))	N/A	N/A		N/A	
☐ EXAMINATION FEE (37 CFR 1.16(o), (p), or (q))	N/A	N/A		N/A	
TOTAL CLAIMS (37 CFR 1.16(i))	minus 20 =	*		x $40 =	
INDEPENDENT CLAIMS (37 CFR 1.16(h))	minus 3 =	*		x $210 =	
☐ APPLICATION SIZE FEE (37 CFR 1.16(s))	If the specification and drawings exceed 100 sheets of paper, the application size fee due is $310 ($155 for small entity) for each additional 50 sheets or fraction thereof. See 35 U.S.C. 41(a)(1)(G) and 37 CFR 1.16(s).				
☐ MULTIPLE DEPENDENT CLAIM PRESENT (37 CFR 1.16(j))					
* If the difference in column 1 is less than zero, enter "0" in column 2.				TOTAL	

APPLICATION AS AMENDED - PART II

		(Column 1)		(Column 2)	(Column 3)			
AMENDMENT	06/17/2019	CLAIMS REMAINING AFTER AMENDMENT		HIGHEST NUMBER PREVIOUSLY PAID FOR	PRESENT EXTRA		RATE ($)	ADDITIONAL FEE ($)
AMENDMENT	Total (37 CFR 1.16(i))	* 20	Minus	** 20	= 0		x $50 =	0
AMENDMENT	Independent (37 CFR 1.16(h))	* 3	Minus	*** 3	= 0		x $230 =	0
AMENDMENT	☐ Application Size Fee (37 CFR 1.16(s))							
AMENDMENT	☐ FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))							
							TOTAL ADD'L FEE	0

		(Column 1)		(Column 2)	(Column 3)			
AMENDMENT		CLAIMS REMAINING AFTER AMENDMENT		HIGHEST NUMBER PREVIOUSLY PAID FOR	PRESENT EXTRA		RATE ($)	ADDITIONAL FEE ($)
AMENDMENT	Total (37 CFR 1.16(i))	*	Minus	**	=		x $0 =	
AMENDMENT	Independent (37 CFR 1.16(h))	*	Minus	***	=		x $0 =	
AMENDMENT	☐ Application Size Fee (37 CFR 1.16(s))							
AMENDMENT	☐ FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))							
							TOTAL ADD'L FEE	

* If the entry in column 1 is less than the entry in column 2, write "0" in column 3.	LIE
** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, enter "20".	/ROLITA M WIMBUSH/
*** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, enter "3".	
The "Highest Number Previously Paid For" (Total or Independent) is the highest number found in the appropriate box in column 1.	

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Inventor:	Jasmin Cosic	Confirmation No.: 1993
Title:	"ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION"	
Serial No.:	15/340,991	Filed: November 2, 2016
Examiner:	GARCIA, SANTIAGO	Group Art Unit: 2668

Via EFS-Web June 17, 2019

Mail Stop Amendment
COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, VA 22313-1450

PRELIMINARY AMENDMENT

Dear Commissioner:

Before examination on the merits, please amend the claims in the above-identified application as shown in the following listing of claims. This preliminary amendment supersedes the preliminary amendments filed on May 4, 2019 and May 27, 2019, therefore, the claims in the following listing of claims should be used for the examination on the merits instead of the claims in the preliminary amendments filed on May 4, 2019 and May 27, 2019.

Listing of Claims

1 - 60 (canceled)

61. (new) A method implemented using a computing system that includes one or more processor circuits, the method comprising:

accessing a memory that stores at least a first correlation including a first one or more object representations correlated with a first one or more instruction sets for operating a first device, wherein the first one or more object representations represent a first one or more objects detected at least in part by one or more sensors of the first device;

generating a second one or more object representations, wherein the second one or more object representations represent a second one or more objects;

anticipating the first one or more instruction sets for operating the first device based on at least partial match between the second one or more object representations and the first one or more object representations; and

at least in response to the determining, executing the first one or more instruction sets for operating the first device, wherein the first device or a second device performs one or more operations defined by the first one or more instruction sets for operating the first device.

62. (new) The method of claim 61, wherein the executing the first one or more instruction sets for operating the first device is performed via an execution interface.

63. (new) The method of claim 61, wherein the first one or more instruction sets for operating the first device include one or more instruction sets for operating a portion of the first device, and wherein the performing, by the first device or by the second device, the one or more operations defined by the first one or more instruction sets for operating the first device includes performing, by the portion of the first device or by a portion of the second device, one or more operations defined by the one or more instruction sets for operating the portion of the first device.

64. (new) The method of claim 61, wherein the first one or more instruction sets for operating the first device are obtained via an acquisition interface.

65. (new) The method of claim 61, wherein at least one selected from the group comprising: the one or more processor circuits, and the memory are included in a remote computing device.

66. (new) The method of claim 61, wherein at least one selected from the group comprising: the one or more processor circuits, and the memory are included in multiple devices.

67. (new) The method of claim 61, wherein at least one selected from the group comprising: the one or more processor circuits, and the memory are included in a single device.

68. (new) The method of claim 61, wherein at least one object of the first one or more objects and at least one object of the second one or more objects are the same.

69. (new) The method of claim 61, wherein the second one or more objects include one or more objects detected at least in part by: the one or more sensors of the first device, or one or more sensors of the second device.

70. (new) The method of claim 61, wherein the one or more processor circuits include: one or more microcontrollers, one or more computing circuits, or one or more electronic circuits, and wherein the memory includes: a volatile memory or a non-volatile memory.

71. (new) One or more non-transitory machine readable media storing machine readable code that when executed by one or more processor circuits causes the one or more processor circuits to perform at least:

accessing a memory that stores at least a first correlation including a first one or more object representations correlated with a first one or more instruction

sets for operating a first device, wherein the first one or more object representations represent a first one or more objects detected at least in part by one or more sensors of the first device;

generating a second one or more object representations, wherein the second one or more object representations represent a second one or more objects;

determining the first one or more instruction sets for operating the first device based on at least partial match between the second one or more object representations and the first one or more object representations; and

at least in response to the determining, causing the first device or a second device to perform one or more operations defined by the first one or more instruction sets for operating the first device at least by causing the one or more processor circuits or another one or more processor circuits to execute the first one or more instruction sets for operating the first device.

72. (new) The one or more non-transitory machine readable media of claim 71, wherein the first one or more instruction sets for operating the first device include one or more instruction sets for operating a portion of the first device, and wherein the performing, by the first device or by the second device, the one or more operations defined by the first one or more instruction sets for operating the first device includes performing, by the portion of the first device or by a portion of the second device, one or more operations defined by the one or more instruction sets for operating the portion of the first device.

73. (new) The one or more non-transitory machine readable media of claim 71, wherein the first one or more instruction sets for operating the first device are obtained via an acquisition interface.

74. (new) The one or more non-transitory machine readable media of claim 71, wherein at least one selected from the group comprising: the one or more processor circuits, the another one or more processor circuits, and the memory are included in a remote computing device.

75. (new) The one or more non-transitory machine readable media of claim 71, wherein at least one selected from the group comprising: the one or more processor circuits, the another one or more processor circuits, and the memory are included in multiple devices.

76. (new) The one or more non-transitory machine readable media of claim 71, wherein at least one selected from the group comprising: the one or more processor circuits, the another one or more processor circuits, and the memory are included in a single device.

77. (new) The one or more non-transitory machine readable media of claim 71, wherein at least one object of the first one or more objects and at least one object of the second one or more objects are the same.

78. (new) The one or more non-transitory machine readable media of claim 71, wherein the second one or more objects include one or more objects detected at least in part by: the one or more sensors of the first device, or one or more sensors of the second device.

79. (new) A system comprising:

a memory that stores at least a first correlation including a first one or more object representations correlated with a first one or more instruction sets for operating a first device, wherein the first one or more object representations represent a first one or more objects detected at least in part by one or more sensors of the first device;

means for generating a second one or more object representations, wherein the second one or more object representations represent a second one or more objects;

means for determining the first one or more instruction sets for operating the first device based on at least partial match between the second one or more object representations and the first one or more object representations; and

means for executing the first one or more instruction sets for operating the first device at least in response to the determining, wherein the first device or a second device performs one or more operations defined by the first one or more instruction sets for operating the first device.

80. (new) The system of claim 79, wherein the means for generating the second one or more object representations includes one or more processor circuits, and wherein the means for determining the first one or more instruction sets for operating the first device based on the at least partial match between the second one or more object representations and the first one or more object representations includes one or more processor circuits, and wherein the means for executing the first one or more instruction sets for operating the first device includes one or more processor circuits.

Remarks

In this preliminary amendment, the applicant cancels claims 1-60, and presents for examination new claims 61-80. After entry of this preliminary amendment, claims 61-80 are pending. The undersigned may be contacted at (317) 772-1312 concerning this application.

I hereby certify that this correspondence is being submitted electronically via EFS Web to the United States Patent and Trademark Office.

By /Jasmin Cosic/
Jasmin Cosic

Date submitted: June 17, 2019

Respectfully submitted,

/Jasmin Cosic/

Jasmin Cosic

Electronic Acknowledgement Receipt

EFS ID:	36322513
Application Number:	15340991
International Application Number:	
Confirmation Number:	1993
Title of Invention:	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION
First Named Inventor/Applicant Name:	Jasmin Cosic
Customer Number:	116094
Filer:	Jasmin Cosic
Filer Authorized By:	
Attorney Docket Number:	
Receipt Date:	17-JUN-2019
Filing Date:	02-NOV-2016
Time Stamp:	17:21:14
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	no

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1		PRELIMINARY_AMENDMENT.pdf	34172 cc223f3bf2cb26a66d7d38330880f1f68ed85184	yes	9

	Multipart Description/PDF files in .zip description		
	Document Description	**Start**	**End**
	Preliminary Amendment	1	1
	Claims	2	8
	Applicant Arguments/Remarks Made in an Amendment	9	9
Warnings:			
Information:			
	Total Files Size (in bytes):	34172	

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.
National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.
New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

NOTICE OF ALLOWANCE AND FEE(S) DUE

116094 7590 06/24/2019

Jasmin Cosic
108 Woodbury Street
Pawtucket, RI 02861

EXAMINER	
GARCIA, SANTIAGO	
ART UNIT	PAPER NUMBER
2668	

DATE MAILED: 06/24/2019

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
15/340,991	11/02/2016	Jasmin Cosic		1993

TITLE OF INVENTION: ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION

APPLN. TYPE	ENTITY STATUS	ISSUE FEE DUE	PUBLICATION FEE DUE	PREV. PAID ISSUE FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	SMALL	$500	$0.00	$0.00	$500	09/24/2019

THE APPLICATION IDENTIFIED ABOVE HAS BEEN EXAMINED AND IS ALLOWED FOR ISSUANCE AS A PATENT. PROSECUTION ON THE MERITS IS CLOSED. THIS NOTICE OF ALLOWANCE IS NOT A GRANT OF PATENT RIGHTS. THIS APPLICATION IS SUBJECT TO WITHDRAWAL FROM ISSUE AT THE INITIATIVE OF THE OFFICE OR UPON PETITION BY THE APPLICANT. SEE 37 CFR 1.313 AND MPEP 1308.

THE ISSUE FEE AND PUBLICATION FEE (IF REQUIRED) MUST BE PAID WITHIN THREE MONTHS FROM THE MAILING DATE OF THIS NOTICE OR THIS APPLICATION SHALL BE REGARDED AS ABANDONED. THIS STATUTORY PERIOD CANNOT BE EXTENDED. SEE 35 U.S.C. 151. THE ISSUE FEE DUE INDICATED ABOVE DOES NOT REFLECT A CREDIT FOR ANY PREVIOUSLY PAID ISSUE FEE IN THIS APPLICATION. IF AN ISSUE FEE HAS PREVIOUSLY BEEN PAID IN THIS APPLICATION (AS SHOWN ABOVE), THE RETURN OF PART B OF THIS FORM WILL BE CONSIDERED A REQUEST TO REAPPLY THE PREVIOUSLY PAID ISSUE FEE TOWARD THE ISSUE FEE NOW DUE.

HOW TO REPLY TO THIS NOTICE:

I. Review the ENTITY STATUS shown above. If the ENTITY STATUS is shown as SMALL or MICRO, verify whether entitlement to that entity status still applies.

If the ENTITY STATUS is the same as shown above, pay the TOTAL FEE(S) DUE shown above.

If the ENTITY STATUS is changed from that shown above, on PART B - FEE(S) TRANSMITTAL, complete section number 5 titled "Change in Entity Status (from status indicated above)".

For purposes of this notice, small entity fees are 1/2 the amount of undiscounted fees, and micro entity fees are 1/2 the amount of small entity fees.

II. PART B - FEE(S) TRANSMITTAL, or its equivalent, must be completed and returned to the United States Patent and Trademark Office (USPTO) with your ISSUE FEE and PUBLICATION FEE (if required). If you are charging the fee(s) to your deposit account, section "4b" of Part B - Fee(s) Transmittal should be completed and an extra copy of the form should be submitted. If an equivalent of Part B is filed, a request to reapply a previously paid issue fee must be clearly made, and delays in processing may occur due to the difficulty in recognizing the paper as an equivalent of Part B.

III. All communications regarding this application must give the application number. Please direct all communications prior to issuance to Mail Stop ISSUE FEE unless advised to the contrary.

IMPORTANT REMINDER: Maintenance fees are due in utility patents issuing on applications filed on or after Dec. 12, 1980. It is patentee's responsibility to ensure timely payment of maintenance fees when due. More information is available at www.uspto.gov/PatentMaintenanceFees.

PART B - FEE(S) TRANSMITTAL

Complete and send this form, together with applicable fee(s), by mail or fax, or via EFS-Web.

By mail, send to: Mail Stop ISSUE FEE
Commissioner for Patents
P.O. Box 1450
Alexandria, Virginia 22313-1450

By fax, send to: (571)-273-2885

INSTRUCTIONS: This form should be used for transmitting the ISSUE FEE and PUBLICATION FEE (if required). Blocks 1 through 5 should be completed where appropriate. All further correspondence including the Patent, advance orders and notification of maintenance fees will be mailed to the current correspondence address as indicated unless corrected below or directed otherwise in Block 1, by (a) specifying a new correspondence address; and/or (b) indicating a separate "FEE ADDRESS" for maintenance fee notifications.

CURRENT CORRESPONDENCE ADDRESS (Note: Use Block 1 for any change of address)

116094 7590 06/24/2019

Jasmin Cosic
108 Woodbury Street
Pawtucket, RI 02861

Note: A certificate of mailing can only be used for domestic mailings of the Fee(s) Transmittal. This certificate cannot be used for any other accompanying papers. Each additional paper, such as an assignment or formal drawing, must have its own certificate of mailing or transmission.

Certificate of Mailing or Transmission

I hereby certify that this Fee(s) Transmittal is being deposited with the United States Postal Service with sufficient postage for first class mail in an envelope addressed to the Mail Stop ISSUE FEE address above, or being transmitted to the USPTO via EFS-Web or by facsimile to (571) 273-2885, on the date below.

(Typed or printed name)

(Signature)

(Date)

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
15/340,991	11/02/2016	Jasmin Cosic		1993

TITLE OF INVENTION: ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION

APPLN. TYPE	ENTITY STATUS	ISSUE FEE DUE	PUBLICATION FEE DUE	PREV. PAID ISSUE FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	SMALL	$500	$0.00	$0.00	$500	09/24/2019

EXAMINER	ART UNIT	CLASS-SUBCLASS
GARCIA, SANTIAGO	2668	382-155000

1. Change of correspondence address or indication of "Fee Address" (37 CFR 1.363).

❑ Change of correspondence address (or Change of Correspondence Address form PTO/SB/122) attached.

❑ "Fee Address" indication (or "Fee Address" Indication form PTO/SB/47; Rev 03-09 or more recent) attached. **Use of a Customer Number is required.**

2. For printing on the patent front page, list
(1) The names of up to 3 registered patent attorneys or agents OR, alternatively,
(2) The name of a single firm (having as a member a registered attorney or agent) and the names of up to 2 registered patent attorneys or agents. If no name is listed, no name will be printed.

1 ____________________
2 ____________________
3 ____________________

3. ASSIGNEE NAME AND RESIDENCE DATA TO BE PRINTED ON THE PATENT (print or type)

PLEASE NOTE: Unless an assignee is identified below, no assignee data will appear on the patent. If an assignee is identified below, the document must have been previously recorded, or filed for recordation, as set forth in 37 CFR 3.11 and 37 CFR 3.81(a). Completion of this form is NOT a substitute for filing an assignment.

(A) NAME OF ASSIGNEE

(B) RESIDENCE: (CITY and STATE OR COUNTRY)

Please check the appropriate assignee category or categories (will not be printed on the patent) : ❑ Individual ❑ Corporation or other private group entity ❑ Government

4a. Fees submitted: ❑Issue Fee ❑Publication Fee (if required) ❑Advance Order - # of Copies ____________

4b. Method of Payment: *(Please first reapply any previously paid fee shown above)*

❑ Electronic Payment via EFS-Web ❑ Enclosed check ❑ Non-electronic payment by credit card (Attach form PTO-2038)

❑ The Director is hereby authorized to charge the required fee(s), any deficiency, or credit any overpayment to Deposit Account No. ________

5. **Change in Entity Status** (from status indicated above)

❑ Applicant certifying micro entity status. See 37 CFR 1.29

❑ Applicant asserting small entity status. See 37 CFR 1.27

❑ Applicant changing to regular undiscounted fee status.

NOTE: Absent a valid certification of Micro Entity Status (see forms PTO/SB/15A and 15B), issue fee payment in the micro entity amount will not be accepted at the risk of application abandonment.
NOTE: If the application was previously under micro entity status, checking this box will be taken to be a notification of loss of entitlement to micro entity status.
NOTE: Checking this box will be taken to be a notification of loss of entitlement to small or micro entity status, as applicable.

NOTE: This form must be signed in accordance with 37 CFR 1.31 and 1.33. See 37 CFR 1.4 for signature requirements and certifications.

Authorized Signature ______________________ Date ______________________

Typed or printed name ______________________ Registration No. ______________________

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
15/340,991	11/02/2016	Jasmin Cosic		1993

116094 7590 06/24/2019

Jasmin Cosic
108 Woodbury Street
Pawtucket, RI 02861

EXAMINER	
GARCIA, SANTIAGO	
ART UNIT	PAPER NUMBER
2668	

DATE MAILED: 06/24/2019

Determination of Patent Term Adjustment under 35 U.S.C. 154 (b)
(Applications filed on or after May 29, 2000)

The Office has discontinued providing a Patent Term Adjustment (PTA) calculation with the Notice of Allowance.

Section 1(h)(2) of the AIA Technical Corrections Act amended 35 U.S.C. 154(b)(3)(B)(i) to eliminate the requirement that the Office provide a patent term adjustment determination with the notice of allowance. See Revisions to Patent Term Adjustment, 78 Fed. Reg. 19416, 19417 (Apr. 1, 2013). Therefore, the Office is no longer providing an initial patent term adjustment determination with the notice of allowance. The Office will continue to provide a patent term adjustment determination with the Issue Notification Letter that is mailed to applicant approximately three weeks prior to the issue date of the patent, and will include the patent term adjustment on the patent. Any request for reconsideration of the patent term adjustment determination (or reinstatement of patent term adjustment) should follow the process outlined in 37 CFR 1.705.

Any questions regarding the Patent Term Extension or Adjustment determination should be directed to the Office of Patent Legal Administration at (571)-272-7702. Questions relating to issue and publication fee payments should be directed to the Customer Service Center of the Office of Patent Publication at 1-(888)-786-0101 or (571)-272-4200.

OMB Clearance and PRA Burden Statement for PTOL-85 Part B

The Paperwork Reduction Act (PRA) of 1995 requires Federal agencies to obtain Office of Management and Budget approval before requesting most types of information from the public. When OMB approves an agency request to collect information from the public, OMB (i) provides a valid OMB Control Number and expiration date for the agency to display on the instrument that will be used to collect the information and (ii) requires the agency to inform the public about the OMB Control Number's legal significance in accordance with 5 CFR 1320.5(b).

The information collected by PTOL-85 Part B is required by 37 CFR 1.311. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 30 minutes to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, Virginia 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, Virginia 22313-1450. Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether disclosure of these records is required by the Freedom of Information Act.
2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.
3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.
4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).
5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.
6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).
7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.
8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspection or an issued patent.
9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Notice of Allowability	Application No. 15/340,991	Applicant(s) Cosic, Jasmin	
	Examiner SANTIAGO GARCIA	Art Unit 2668	AIA (FITF) Status Yes

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address--

All claims being allowable, PROSECUTION ON THE MERITS IS (OR REMAINS) CLOSED in this application. If not included herewith (or previously mailed), a Notice of Allowance (PTOL-85) or other appropriate communication will be mailed in due course. **THIS NOTICE OF ALLOWABILITY IS NOT A GRANT OF PATENT RIGHTS.** This application is subject to withdrawal from issue at the initiative of the Office or upon petition by the applicant. See 37 CFR 1.313 and MPEP 1308.

1.☑ This communication is responsive to 06/17/19 last set of claims submitted.
☐ A declaration(s)/affidavit(s) under **37 CFR 1.130(b)** was/were filed on ______.

2.☐ An election was made by the applicant in response to a restriction requirement set forth during the interview on ______; the restriction requirement and election have been incorporated into this action.

3.☑ The allowed claim(s) is/are 61-80 . As a result of the allowed claim(s), you may be eligible to benefit from the **Patent Prosecution Highway** program at a participating intellectual property office for the corresponding application. For more information, please see **http://www.uspto.gov/patents/init_events/pph/index.jsp** or send an inquiry to **PPHfeedback@uspto.gov.**

4.☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

Certified copies:

a) ☐All b) ☐ Some *c) ☐ None of the:

1. ☐ Certified copies of the priority documents have been received.
2. ☐ Certified copies of the priority documents have been received in Application No. ______ .
3. ☐ Copies of the certified copies of the priority documents have been received in this national stage application from the International Bureau (PCT Rule 17.2(a)).

* Certified copies not received: ______ .

Applicant has THREE MONTHS FROM THE "MAILING DATE" of this communication to file areply complying with the requirements noted below. Failure to timely comply will result in ABANDONMENT of this application.
THIS THREE-MONTH PERIOD IS NOT EXTENDABLE.

5.☐ CORRECTED DRAWINGS (as "replacement sheets") must be submitted.
☐ including changes required by the attached Examiner's Amendment / Comment or in the Office action of Paper No./Mail Date ______ .

Identifying indicia such as the application number (see 37 CFR 1.84(c)) should be written on the drawings in the front (not the back) of each sheet. Replacement sheet(s) should be labeled as such in the header according to 37 CFR 1.121(d).

6.☐ DEPOSIT OF and/or INFORMATION about the deposit of BIOLOGICAL MATERIAL must be submitted. Note the attached Examiner's comment regarding REQUIREMENT FOR THE DEPOSIT OF BIOLOGICAL MATERIAL.

Attachment(s)

1.☑ Notice of References Cited (PTO-892)
2.☑ Information Disclosure Statements (PTO/SB/08), Paper No./Mail Date ______.
3.☐ Examiner's Comment Regarding Requirement for Deposit of Biological Material ______.
4.☐ Interview Summary (PTO-413), Paper No./Mail Date. ______.
5. ☑ Examiner's Amendment/Comment
6. ☑ Examiner's Statement of Reasons for Allowance
7. ☐ Other ______.

/SANTIAGO GARCIA/
Primary Examiner, Art Unit 2668

Notice of Pre-AIA or AIA Status

The present application, filed on or after March 16, 2013, is being examined under the first inventor to file provisions of the AIA.

Claim Interpretation

The following is a quotation of 35 U.S.C. 112(f):

> (f) Element in Claim for a Combination. – An element in a claim for a combination may be expressed as a means or step for performing a specified function without the recital of structure, material, or acts in support thereof, and such claim shall be construed to cover the corresponding structure, material, or acts described in the specification and equivalents thereof.

The following is a quotation of pre-AIA 35 U.S.C. 112, sixth paragraph:

> An element in a claim for a combination may be expressed as a means or step for performing a specified function without the recital of structure, material, or acts in support thereof, and such claim shall be construed to cover the corresponding structure, material, or acts described in the specification and equivalents thereof.

The claims in this application are given their broadest reasonable interpretation using the plain meaning of the claim language in light of the specification as it would be understood by one of ordinary skill in the art. The broadest reasonable interpretation of a claim element (also commonly referred to as a claim limitation) is limited by the description in the specification when 35 U.S.C. 112(f) or pre-AIA 35 U.S.C. 112, sixth paragraph, is invoked.

As explained in MPEP § 2181, subsection I, claim limitations that meet the following three-prong test will be interpreted under 35 U.S.C. 112(f) or pre-AIA 35 U.S.C. 112, sixth paragraph:

(A) the claim limitation uses the term “means” or “step” or a term used as a substitute for “means” that is a generic placeholder (also called a nonce term or a non-structural term having no specific structural meaning) for performing the claimed function;

(B) the term “means” or “step” or the generic placeholder is modified by functional language, typically, but not always linked by the transition word “for” (e.g., “means for”) or another linking word or phrase, such as “configured to” or “so that”; and

(C) the term “means” or “step” or the generic placeholder is not modified by sufficient structure, material, or acts for performing the claimed function.

Use of the word “means” (or “step”) in a claim with functional language creates a rebuttable presumption that the claim limitation is to be treated in accordance with 35 U.S.C. 112(f) or pre-AIA 35 U.S.C. 112, sixth paragraph. The presumption that the claim limitation is interpreted under 35 U.S.C. 112(f) or pre-AIA 35 U.S.C. 112, sixth paragraph, is rebutted when the claim limitation recites sufficient structure, material, or acts to entirely perform the recited function.

Absence of the word “means” (or “step”) in a claim creates a rebuttable presumption that the claim limitation is not to be treated in accordance with 35 U.S.C. 112(f) or pre-AIA 35 U.S.C. 112, sixth paragraph. The presumption that the claim limitation is not interpreted under 35 U.S.C. 112(f) or pre-AIA 35 U.S.C. 112, sixth paragraph, is rebutted when the claim limitation recites function without reciting sufficient structure, material or acts to entirely perform the recited function.

Claim limitations 79-80 in this application that use the word “means” (or “step”) are being interpreted under 35 U.S.C. 112(f) or pre-AIA 35 U.S.C. 112, sixth paragraph, except as otherwise indicated in an Office action. Conversely, claim limitations in this application that do not use the word “means” (or “step”) are not being interpreted under 35 U.S.C. 112(f) or pre-AIA 35 U.S.C. 112, sixth paragraph, except as otherwise indicated in an Office action.

EXAMINER'S AMENDMENT

An examiner's amendment to the record appears below. Should the changes and/or additions be unacceptable to applicant, an amendment may be filed as provided by 37 CFR 1.312. To ensure consideration of such an amendment, it MUST be submitted no later than the payment of the issue fee.

Authorization for this examiner's amendment was given in an interview with Jasmin, Cosic on 6/18/19.

All claims 61-80 are amended as follows, all claims are currently amended.

The application has been amended as follows:

61. A method implemented using a computing system that includes one or more processor circuits, the method comprising:

accessing a memory that stores at least a knowledgebase that includes: a first correlation including a first circumstance representation correlated with a first one or more instruction sets for operating a first device and a second correlation including a second circumstance representation correlated with a second one or more instruction sets for operating the first device, wherein the first circumstance representation represents a first circumstance detected at least in part by one or more sensors of the first device and the second circumstance representation represents a second circumstance detected at least in part by the one or more sensors of the first device, and wherein at least a portion of the first correlation and at least a portion of the second correlation are learned in a learning process that includes operating the first device at least partially by a user;

generating or receiving a third circumstance representation, wherein the third circumstance representation represents a third circumstance detected at least in part by the one or more sensors of the first device or at least in part by one or more sensors of a second device;

anticipating the first one or more instruction sets for operating the first device learned in the learning process based on at least partial match between the third circumstance representation and the first circumstance representation; and

at least in response to the anticipating, executing the first one or more instruction sets for operating the first device learned in the learning process, wherein the first device or [[a]] the second device autonomously performs one or more operations defined by the first one or more instruction sets for operating the first device learned in the learning process.

62. The method of claim 61, wherein the knowledgebase further includes: a first knowledge cell and a second knowledge cell, and wherein the first knowledge cell includes the first correlation and the second knowledge cell includes the second correlation.

63. The method of claim 61, wherein the learning process includes: creating, inserting, deleting, modifying, or manipulating an element of the first correlation, or creating, inserting, deleting, modifying, or manipulating an element of the second correlation.

64. The method of claim 61, wherein the learning process includes:

generating or receiving the first circumstance representation, and generating or receiving the second circumstance representation; and

obtaining or receiving the first one or more instruction sets for operating the first device, and obtaining or receiving the second one or more instruction sets for operating the first device, and wherein the knowledgebase further includes a fourth correlation including a fourth circumstance representation correlated with a fourth one or more instruction sets for operating the second device, and wherein the fourth circumstance representation represents a fourth circumstance detected at least

in part by the one or more sensors of the second device, and wherein at least a portion of the fourth correlation is learned in another learning process that includes:

generating or receiving the fourth circumstance representation; and

obtaining or receiving the fourth one or more instruction sets for operating the second device.

65. The method of claim 61, wherein the anticipating the first one or more instruction sets for operating the first device learned in the learning process based on the at least partial match between the third circumstance representation and the first circumstance representation includes:

determining that a number of at least partially matching portions of the third circumstance representation and portions of the first circumstance representation exceeds a threshold number; or

determining that a percentage of at least partially matching portions of the third circumstance representation and portions of the first circumstance representation exceeds a threshold percentage.

66. The method of claim 61, wherein the at least the portion of the first correlation and the at least the portion of the second correlation are learned in the learning process while the user operates the first device, and wherein the one or more operations defined by the first one or more instruction sets for operating the first device learned in the learning process correspond to the user's methodology of operating the first device learned in the learning process.

67. The method of claim 61, wherein the knowledgebase further includes a fourth correlation including a fourth circumstance representation correlated with a fourth one or more instruction sets for operating: the first device, the second device, or a third device, and wherein a first connection is generated to connect the first correlation with the second correlation, and wherein a second connection is generated to connect the second correlation with the fourth correlation.

68. The method of claim 61, further comprising:

modifying the first one or more instruction sets for operating the first device learned in the learning process or a copy of the first one or more instruction sets for operating the first device learned in the learning process, and wherein the executing the first one or more instruction sets for operating the first device learned in the learning process includes executing the modified the first one or more instruction sets for operating the first device learned in the learning process or the modified the copy of the first one or more instruction sets for operating the first device learned in the learning process, and wherein the performing, by the first device or by the second device, the one or more operations defined by the first one or more instruction sets for operating the first device learned in the learning process includes performing, by the first device or by the second device, one or more operations defined by the modified the first one or more instruction sets for operating the first device learned in the learning process or by the modified the copy of the first one or more instruction sets for operating the first device learned in the learning process.

69. The method of claim 61, wherein the knowledgebase further includes a fourth correlation including a fourth circumstance representation correlated with a fourth one or more instruction sets for operating the first device, and wherein the fourth circumstance representation represents a fourth circumstance detected at least in part by the one or more sensors of the first device, and wherein at

least a portion of the fourth correlation is learned in another learning process that includes operating the first device at least partially by the user.

70. The method of claim 61, wherein the knowledgebase further includes a fourth correlation including a fourth circumstance representation correlated with a fourth one or more instruction sets for operating the first device, and wherein the fourth circumstance representation represents a fourth circumstance detected at least in part by the one or more sensors of the first device, and wherein at least a portion of the fourth correlation is learned in another learning process that includes operating the first device at least partially by another user.

71. The method of claim 61, wherein the knowledgebase further includes a fourth correlation including a fourth circumstance representation correlated with a fourth one or more instruction sets for operating the second device, and wherein the fourth circumstance representation represents a fourth circumstance detected at least in part by the one or more sensors of the second device, and wherein at least a portion of the fourth correlation is learned in another learning process that includes operating the second device at least partially by another user.

72. The method of claim 61, wherein the first circumstance includes one or more objects detected at least in part by the one or more sensors of the first device at a first time or during a first time period, and wherein the second circumstance includes one or more objects detected at least in part by the one or more sensors of the first device at a second time or during a second time period, and wherein the third circumstance includes:

one or more objects detected at least in part by the one or more sensors of the first device at a third time or during a third time period, or

one or more objects detected at least in part by the one or more sensors of the second device at a third time or during a third time period.

73. The method of claim 61, wherein the first circumstance representation is a data structure including one or more data about the first circumstance of the first device, and wherein the second circumstance representation is a data structure including one or more data about the second circumstance of the first device, and wherein the third circumstance representation is a data structure including one or more data about: the third circumstance of the first device or the third circumstance of the second device.

74. The method of claim 61, wherein the first circumstance representation includes: one or more object representations, or one or more collections of object representations, and wherein the second circumstance representation includes: one or more object representations, or one or more collections of object representations, and wherein the third circumstance representation includes: one or more object representations, or one or more collections of object representations.

75. The method of claim 61, wherein, to correlate the first circumstance representation with the first one or more instruction sets for operating the first device, a determination is made that the first one or more instruction sets for operating the first device temporally correspond to the first circumstance representation, wherein the first circumstance representation includes: one or more object representations, or one or more collections of object representations.

76. The method of claim 61, wherein the anticipating the first one or more instruction sets for operating the first device learned in the learning process based on the at least partial match between

the third circumstance representation and the first circumstance representation includes determining the first one or more instruction sets for operating the first device learned in the learning process based on the at least partial match between the third circumstance representation and the first circumstance representation.

77. The method of claim 61, wherein elements of the computing system are included in: a single device, or multiple devices, and wherein the one or more processor circuits include: one or more microcontrollers, one or more computing circuits, or one or more electronic circuits, and wherein the memory includes: a volatile memory, or a non-volatile memory, and wherein the first device includes: a robot, a vehicle, an appliance, an electronic device, or a mechanical machine, and wherein the second device includes: a robot, a vehicle, an appliance, an electronic device, or a mechanical machine, and wherein an instruction set of the first one or more instruction sets for operating the first device includes at least one selected from the group comprising: only one instruction, a plurality of instructions, one or more inputs, one or more commands, one or more computer commands, one or more keywords, one or more symbols, one or more operators, one or more variables, one or more values, one or more objects or object references, one or more data structures or data structure references, one or more functions or function references, one or more parameters, one or more signals, one or more characters, one or more digits, one or more user operating directions, one or more user directions, one or more user inputs, one or more representations of one or more user actions, one or more representations of one or more user clicks, one or more binary bits, one or more assembly language commands, one or more states, one or more codes, one or more data, and one or more information, and wherein an instruction set of the second one or more instruction sets for operating the first device includes at least one selected from the group comprising: only one instruction, a plurality of instructions, one or more inputs, one or more commands, one or more computer commands, one or more keywords, one or more symbols, one or

more operators, one or more variables, one or more values, one or more objects or object references, one or more data structures or data structure references, one or more functions or function references, one or more parameters, one or more signals, one or more characters, one or more digits, one or more user operating directions, one or more user directions, one or more user inputs, one or more representations of one or more user actions, one or more representations of one or more user clicks, one or more binary bits, one or more assembly language commands, one or more states, one or more codes, one or more data, and one or more information, and wherein the one or more sensors of the first device include at least one selected from the group comprising: one or more cameras, one or more microphones, one or more radars, one or more lidars, one or more sonars, and one or more apparatuses for detecting objects or object properties, and wherein the one or more sensors of the second device include at least one selected from the group comprising: one or more cameras, one or more microphones, one or more radars, one or more lidars, one or more sonars, and one or more apparatuses for detecting objects or object properties, and wherein the at least the portion of the first correlation includes: one portion of the first correlation, multiple portions of the first correlation, all portions of the first correlation, or the entire first correlation, and wherein the at least the portion of the second correlation includes: one portion of the second correlation, multiple portions of the second correlation, all portions of the second correlation, or the entire second correlation, and wherein an object of the first circumstance is the same as an object of the third circumstance, or multiple objects of the first circumstance are the same as multiple objects of the third circumstance, or all objects of the first circumstance are the same as all objects of the third circumstance, or all objects of the first circumstance are different than all objects of the third circumstance.

78. One or more non-transitory machine readable media storing machine readable code that when executed by one or more processor circuits causes the one or more processor circuits to perform at least:

accessing a memory that stores at least a knowledgebase that includes: a first correlation including a first circumstance representation correlated with a first one or more instruction sets for operating a first device and a second correlation including a second circumstance representation correlated with a second one or more instruction sets for operating the first device, wherein the first circumstance representation represents a first circumstance detected at least in part by one or more sensors of the first device and the second circumstance representation represents a second circumstance detected at least in part by the one or more sensors of the first device, and wherein at least a portion of the first correlation and at least a portion of the second correlation are learned in a learning process that includes operating the first device at least partially by a user;

generating or receiving a third circumstance representation, wherein the third circumstance representation represents a third circumstance detected at least in part by the one or more sensors of the first device or at least in part by one or more sensors of a second device;

anticipating the first one or more instruction sets for operating the first device learned in the learning process based on at least partial match between the third circumstance representation and the first circumstance representation; and

at least in response to the anticipating, causing the first device or [[a]] the second device to perform one or more operations defined by the first one or more instruction sets for operating the first device learned in the learning process at least by causing the one or more processor circuits or another one or more processor circuits to execute the first one or more instruction sets for operating the first device learned in the learning process.

79. A system comprising:

a memory that stores at least a knowledgebase that includes: a first correlation including a first circumstance representation correlated with a first one or more instruction sets for operating a first device and a second correlation including a second circumstance representation correlated with a second one or more instruction sets for operating the first device, wherein the first circumstance representation represents a first circumstance detected at least in part by one or more sensors of the first device and the second circumstance representation represents a second circumstance detected at least in part by the one or more sensors of the first device, and wherein at least a portion of the first correlation and at least a portion of the second correlation are learned in a learning process that includes operating the first device at least partially by a user;

means for generating or receiving a third circumstance representation, wherein the third circumstance representation represents a third circumstance detected at least in part by the one or more sensors of the first device or at least in part by one or more sensors of a second device;

means for anticipating the first one or more instruction sets for operating the first device learned in the learning process based on at least partial match between the third circumstance representation and the first circumstance representation; and

means for executing, at least in response to the anticipating, the first one or more instruction sets for operating the first device learned in the learning process, wherein the first device or [[a]] the second device autonomously performs one or more operations defined by the first one or more instruction sets for operating the first device learned in the learning process.

80. The system of claim 79, wherein the means for generating or receiving the third circumstance representation includes one or more processor circuits, and wherein the means for anticipating the first one or more instruction sets for operating the first device learned in the learning

process based on the at least partial match between the third circumstance representation and the first circumstance representation includes one or more processor circuits, and wherein the means for executing, at least in response to the anticipating, the first one or more instruction sets for operating the first device learned in the learning process includes one or more processor circuits.

Reasons for Allowance

The following is an examiner's statement of reasons for allowance: Applicant has disclosed an artificial intelligence system that creates a knowledge base based on operations happening by a device and partially used by a user in part. After instruction sets get stored in a training mode by matching up an instances or objects with training sets, correlations happen as signals from sensors start to come in as a devices gets used. These correlations correlate instructions sets that were gathered in the training mode and stored in the knowledge base and the system then expects different actions according to the instruction sets to happen based on these correlations which in part are caused by a user using the device. Then automatically in response to the anticipating, executing the first one or more instruction sets for operating the first device learned in the learning process, wherein the first device or a second device autonomously performs one or more operations defined by the first one or more instruction sets for operating the first device learned in the learning process and that come up in the correlating process and the anticipating step. There are references that claim the broad spectrum idea, however they do not deal with the specifics of the current claim limitations, such as the prior art does not disclose the device that operates as the user directs it, hence, does not disclose learning the device operation from the user directing the device. Prior art discloses a system that explores its own environment on its own, which is very different from relying on the user to direct it. Prior art does not disclose the first and the second correlations that each include a circumstance representation correlated with instruction sets. Prior art

does not disclose the knowledgebase, which includes a 3-level nested data structure recited in the independent claims. Such as individual circumstance representation (a data structure itself) and individual instruction set (a data structure itself when learned in the learning process). And correlation (a data structure itself) that includes a circumstance representation correlated with instruction sets. And knowledgebase a data structure itself including a plurality of correlation data structures in various arrangements. Further autonomous operation of the device in the claims performs the operation and does not rely on the re-training such as in the prior art as noted below, and the re-training is a crucial function in the proper operating of the systems in the prior art. As an example for highlighting these differences Schnittman (US 2016/0167226) in fig.5 505-510 can be consider the training, and determine robot behavior 535 can be consider instruction sets, and 545 can be considered a correlation since it is going to grab images from the buffer and change the database accordingly, then the classifier can be considered the instances. However the automatic portion of the device actually performing the task without the re-training based on those correlations is not in the prior art as an example. There is always a re-training of some sorts, where the current application automatically controls the device based on that prior correlation and no need for re-training of any kind. Many other references could be cited but the same analysis and differences will be noted.

Any comments considered necessary by applicant must be submitted no later than the payment of the issue fee and, to avoid processing delays, should preferably accompany the issue fee. Such submissions should be clearly labeled “Comments on Statement of Reasons for Allowance.”

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to SANTIAGO GARCIA whose telephone number is (571)270-5182. The examiner can normally be reached on Monday-Friday 9:30am-5:30pm.

Examiner interviews are available via telephone, in-person, and video conferencing using a USPTO supplied web-based collaboration tool. To schedule an interview, applicant is encouraged to use the USPTO Automated Interview Request (AIR) at http://www.uspto.gov/interviewpractice.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, **Le, Vu** can be reached on **(571) 272-7332**. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/SANTIAGO GARCIA/
Primary Examiner, Art Unit 2668

/SG/

Notice of References Cited	Application/Control No. 15/340,991	Applicant(s)/Patent Under Reexamination Cosic, Jasmin	
	Examiner SANTIAGO GARCIA	Art Unit 2668	Page 1 of 2

U.S. PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Name	CPC Classification	US Classification
*	A	US-20160243701-A1	08-2016	Gildert; Suzanne	G05B19/42	1/1
*	B	US-20180120813-A1	05-2018	COFFMAN; Valerie R.	G05B19/4097	1/1
*	C	US-20180101146-A1	04-2018	Hariharan; Rajaraman	G05B13/0265	1/1
*	D	US-8595154-B2	11-2013	Breckenridge; Jordan M.	G06N20/00	706/12
*	E	US-20160274187-A1	09-2016	MENON; SANKARAN M.	G01R31/31718	1/1
*	F	US-9367806-B1	06-2016	Cosic; Jasmin	G06N5/022	1/1
*	G	US-20160167226-A1	06-2016	Schnittman; Mark S.	B25J5/00	382/153
*	H	US-9282309-B1	03-2016	Cosic; Jasmin	H04N9/87	1/1
*	I	US-20150339213-A1	11-2015	LEE; Christopher Stephen	G06F11/3664	717/125
*	J	US-20140177946-A1	06-2014	LIM; Kil-Taek	G06K9/4614	382/156
*	K	US-20120290347-A1	11-2012	ELAZOUNI; ASHRAF	G06Q10/00	705/7.12
*	L	US-20120167057-A1	06-2012	Schmich; Christopher P.	G06F11/3644	717/130
*	M	US-20100278420-A1	11-2010	Shet; Vinay Damodar	G06K9/00369	382/156

FOREIGN PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Country	Name	CPC Classification
	N					
	O					
	P					
	Q					
	R					
	S					
	T					

NON-PATENT DOCUMENTS

*		Include as applicable: Author, Title Date, Publisher, Edition or Volume, Pertinent Pages)
	U	
	V	
	W	
	X	

*A copy of this reference is not being furnished with this Office action. (See MPEP § 707.05(a).)
Dates in MM-YYYY format are publication dates. Classifications may be US or foreign.

Notice of References Cited	Application/Control No. 15/340,991	Applicant(s)/Patent Under Reexamination Cosic, Jasmin	
	Examiner SANTIAGO GARCIA	Art Unit 2668	Page 2 of 2

U.S. PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Name	CPC Classification	US Classification
*	A	US-20170161958-A1	06-2017	EILAT; Eran	G02B27/017	1/1
*	B	US-20160077166-A1	03-2016	Morozov; Alexey	G01C19/00	702/150
*	C	US-9443192-B1	09-2016	Cosic; Jasmin	G06N3/02	1/1
*	D	US-6408262-B1	06-2002	Leerberg; Henrik	G06F11/3608	703/2
*	E	US-9864933-B1	01-2018	Cosic; Jasmin	G06K9/66	1/1
*	F	US-10102449-B1	10-2018	Cosic; Jasmin	G06K9/00771	1/1
*	G	US-10102226-B1	10-2018	Cosic; Jasmin	G06F16/5866	1/1
*	H	US-9582762-B1	02-2017	Cosic; Jasmin	G06N5/04	1/1
	I					
	J					
	K					
	L					
	M					

FOREIGN PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Country	Name	CPC Classification
	N					
	O					
	P					
	Q					
	R					
	S					
	T					

NON-PATENT DOCUMENTS

*		Include as applicable: Author, Title Date, Publisher, Edition or Volume, Pertinent Pages)
	U	
	V	
	W	
	X	

*A copy of this reference is not being furnished with this Office action. (See MPEP § 707.05(a).)
Dates in MM-YYYY format are publication dates. Classifications may be US or foreign.

Index of Claims	Application/Control No. 15/340,991	Applicant(s)/Patent Under Reexamination Cosic, Jasmin
	Examiner SANTIAGO GARCIA	Art Unit 2668

✓	Rejected		-	Cancelled		N	Non-Elected		A	Appeal
=	Allowed		÷	Restricted		I	Interference		O	Objected

CLAIMS

☐ Claims renumbered in the same order as presented by applicant ☐ CPA ☐ T.D. ☐ R.1.47

CLAIM		DATE								
Final	Original	06/19/2019								
	1	-								
	2	-								
	3	-								
	4	-								
	5	-								
	6	-								
	7	-								
	8	-								
	9	-								
	10	-								
	11	-								
	12	-								
	13	-								
	14	-								
	15	-								
	16	-								
	17	-								
	18	-								
	19	-								
	20	-								
	21	-								
	22	-								
	23	-								
	24	-								
	25	-								
	26	-								
	27	-								
	28	-								
	29	-								
	30	-								
	31	-								
	32	-								
	33	-								
	34	-								
	35	-								
	36	-								
	37	-								
	38	-								
	39	-								
	40	-								
	41	-								
	42	-								

Index of Claims	Application/Control No. 15/340,991	Applicant(s)/Patent Under Reexamination Cosic, Jasmin
	Examiner SANTIAGO GARCIA	Art Unit 2668

CLAIM		DATE								
Final	Original	06/19/2019								
	43	-								
	44	-								
	45	-								
	46	-								
	47	-								
	48	-								
	49	-								
	50	-								
	51	-								
	52	-								
	53	-								
	54	-								
	55	-								
	56	-								
	57	-								
	58	-								
	59	-								
	60	-								
	61	=								
	62	=								
	63	=								
	64	=								
	65	=								
	66	=								
	67	=								
	68	=								
	69	=								
	70	=								
	71	=								
	72	=								
	73	=								
	74	=								
	75	=								
	76	=								
	77	=								
	78	=								
	79	=								
	80	=								

Issue Classification	Application/Control No. 15/340,991	Applicant(s)/Patent Under Reexamination Cosic, Jasmin
	Examiner SANTIAGO GARCIA	Art Unit 2668

CPC						
Symbol					**Type**	**Version**
G06N	/	3	/	08	F	2013-01-01
G06N	/	5	/	022	I	2013-01-01

CPC Combination Sets								
Symbol					**Type**	**Set**	**Ranking**	**Version**
	/		/					

NONE (Assistant Examiner) (Date)	**Total Claims Allowed:** 20	
/SANTIAGO GARCIA/ Primary Examiner, Art Unit 2668 19 June 2019 (Primary Examiner) (Date)	O.G. Print Claim(s) 1	O.G. Print Figure 12

U.S. Patent and Trademark Office Part of Paper No.: 20190619

Issue Classification	Application/Control No.	Applicant(s)/Patent Under Reexamination
	15/340,991	Cosic, Jasmin
	Examiner	Art Unit
	SANTIAGO GARCIA	2668

INTERNATIONAL CLASSIFICATION				
CLAIMED				
G06N3/08	/	3	/	08
G06N5/02	/	5	/	02

NON-CLAIMED				
	/		/	

US ORIGINAL CLASSIFICATION	
CLASS	**SUBCLASS**

CROSS REFERENCES(S)						
CLASS	**SUBCLASS (ONE SUBCLASS PER BLOCK)**					

NONE (Assistant Examiner)	(Date)	**Total Claims Allowed:** 20	
/SANTIAGO GARCIA/ Primary Examiner, Art Unit 2668 (Primary Examiner)	19 June 2019 (Date)	O.G. Print Claim(s) 1	O.G. Print Figure 12

U.S. Patent and Trademark Office

Part of Paper No.: 20190619

Issue Classification	Application/Control No. 15/340,991	Applicant(s)/Patent Under Reexamination Cosic, Jasmin
	Examiner SANTIAGO GARCIA	Art Unit 2668

☐ Claims renumbered in the same order as presented by applicant ☐ CPA ☐ T.D. ☐ R.1.47

CLAIMS

Final	Original	Final	Original	Final	Original	Final	Original	Final	Original	Final	Original	Final	Original	Final	Original
	1		11		21		31		41		51	1	61	11	71
	2		12		22		32		42		52	2	62	12	72
	3		13		23		33		43		53	3	63	13	73
	4		14		24		34		44		54	4	64	14	74
	5		15		25		35		45		55	5	65	15	75
	6		16		26		36		46		56	6	66	16	76
	7		17		27		37		47		57	7	67	17	77
	8		18		28		38		48		58	8	68	18	78
	9		19		29		39		49		59	9	69	19	79
	10		20		30		40		50		60	10	70	20	80

NONE (Assistant Examiner) (Date)	**Total Claims Allowed:** 20	
/SANTIAGO GARCIA/ 19 June 2019 Primary Examiner, Art Unit 2668 (Primary Examiner) (Date)	O.G. Print Claim(s) 1	O.G. Print Figure 12

Search Notes	Application/Control No. 15/340,991	Applicant(s)/Patent Under Reexamination Cosic, Jasmin
	Examiner SANTIAGO GARCIA	Art Unit 2668

CPC - Searched*		
Symbol	**Date**	**Examiner**
G06N3/08, G06N 5/022	06/19/2019	SG

CPC Combination Sets - Searched*		
Symbol	**Date**	**Examiner**

US Classification - Searched*			
Class	**Subclass**	**Date**	**Examiner**

* See search history printout included with this form or the SEARCH NOTES box below to determine the scope of the search.

Search Notes		
Search Notes	**Date**	**Examiner**
East and inventor name search	06/19/2019	SG
ip.com, google scholar, IEEE search same search and keywords and synonyms as east search and adopting it to each site for example ((ANTICIPAT$4 OR PREDICT$5 or determ$5) NEAR4 (INSTRUCTION NEAR4 SETS)) with many synonyms for each term along with all the different ways to search for AI	06/19/2019	SG
Consulted with Bernard Krasnic about the invention and [possible areas to search	06/19/2019	SG
Consulted with Niu, Feng with regards to 112f and how to correctly point out the claim interpretation	06/19/2019	SG
EAST search had to be re-created as some search strings were taking too long and work space ended up not working. Re-created as best as could remember.	06/19/2019	SG

/SANTIAGO GARCIA/ Primary Examiner, Art Unit 2668	

Search Notes	Application/Control No.	Applicant(s)/Patent Under Reexamination
	15/340,991	Cosic, Jasmin
	Examiner	**Art Unit**
	SANTIAGO GARCIA	2668

Interference Search			
US Class/CPC Symbol	**US Subclass/CPC Group**	**Date**	**Examiner**
G06N3	08	06/19/2019	SG
G06N5	022	06/19/2019	SG

/SANTIAGO GARCIA/ Primary Examiner, Art Unit 2668	

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

PTO/SB/08a (01-10)
Approved for use through 07/31/2012. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it contains a valid OMB control number.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS Remove

Examiner Initial*	Cite No	Patent Number	Kind Code[1]	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	8996432		2015-03-31	Fu; Jicheng	
	2	6754631		2004-06-22	Din	
	3	9305216		2016-04-05	Mishra	
	4	5560011		1996-09-23	Uyama	
	5	6842877		2005-01-11	Robarts , et al.	
	6	7565340		2009-07-21	Herlocker , et al.	
	7	8261199		2012-09-04	Cradick , et al.	
	8	8266608		2012-09-11	Hecht , et al.	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	9	9268454		2016-02-23	Hamilton, II , et al.	

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS Remove

Examiner Initial*	Cite No	Publication Number	Kind Code1	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	20120290347		2012-11-15	ELAZOUNI; ASHRAF ; et al.	
	2	20150324685		2015-11-12	Bohn; Richard Esten ; et al.	
	3	20100114746		2010-05-06	Bobbitt; Russell Patrick ; et al.	
	4	20120284026		2012-11-08	Cardillo; Peter S. ; et al.	
	5	20140211988		2014-07-31	Fan; Quanfu ; et al.	
	6	20150006171		2015-01-01	Westby; Michael C. ; et al.	
	7	20150264306		2015-09-17	Marilly; Emmanuel ; et al.	
	8	20110085734		2011-04-14	Berg; Jared S. ; et al.	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	9	20150269415		2015-09-24	Gelbman; Alexander	
	10	20160274187		2016-09-22	MENON; SANKARAN M. ; et al.	
	11	20150339213		2015-11-22	LEE; Christopher Stephen ; et al.	
	12	20150310041		2015-10-29	Kier; Scott ; et al.	
	13	20160328480		2016-11-10	Owens; Erich James ; et al.	
	14	20140075249		2014-03-13	SATO; Shuhei ; et al.	
	15	20110270794		2011-11-03	Drory; Tal ; et al.	
	16	20070050719		2007-03-01	Lui; Philip ; et al.	
	17	20070061735		2007-03-15	Hoffberg; Steven M. ; et al.	
	18	20060184410		2006-08-17	Ramamurthy; Shankar ; et al.	
	19	20060265406		2006-11-23	Chkodrov; Gueorgui B. ; et al.	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	20	20090287643		2009-11-19	Corville; Allen O. ; et al.	
	21	20110007079		2011-01-13	Perez et al.	
	22	20070050606		2007-03-01	Ferren et al.	
	23	20090067727		2009-03-12	HIROHATA; Hitoshi	
	24	20130007532		2013-01-03	Miller et al.	
	25	20050240412		2005-10-27	Fujita	
	26	20160167226		2016-06-16	Schnittman	
	27	20090136095		2009-05-28	Marcon et al.	
	28	20080215508		2008-09-04	Hanneman; Jeffrey E. ; et al.	
	29	20090044113		2009-02-12	Jones; Scott T. ; et al.	
	30	20130278501		2013-10-24	BULZACKI; Adrian	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number3	Country Code2 i	Kind Code4	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T5
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T5
	1	Chen et al. "Case-Based Reasoning System and Artificial Neural Networks: A Review Neural Comput & Applic (2001) 10: pp 264-276, 13 pages	
	2	JOHN J. GREFENSTETTE, CONNIE LOGGIA RAMSEY, ALAN C. SCHULTZ, Learning Sequential Decision Rules Using Simulation Models and Competition, 1990, Navy Center for Applied Research in Artificial Intelligence, Naval Research Laboratory, Washington, DC, 27 pages	
	3	ALAN C. SCHULTZ, JOHN J. GREFENSTETTE, Using a Genetic Algorithm to Learn Behaviors for Autonomous Vehicles, 1992, Navy Center for Applied Research in Artificial Intelligence, Naval Research Laboratory, Washington, DC, 12 pages	
	4	Koppula et al., "Anticipating human activities using object affordances for reactive robotic response", IEEE TRAMI 2016, published 5 May 2015, 16 pages	
	5	Orme, "System design tips for entry level smartphones - part 3", found online at "https://community.arm.com/processors/b/blog/posts/system-design-tips-for-entry-level-smartphones---part-3", Oct 21, 2013, 12 pages	
	6	NOWOSTAWSKI, MARIUSZ et al., "Dynamic Demes Parallel Genetic Algorithm", May 13, 1999, 6 pages	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	7		Stack Overflow, How can I quantify difference between two images?, accessed 2 August 2017 at https://stackoverflow.com/questions/189943/how-can-i-quantify-difference-between-two-images, 8 pages	
	8		SKOLICKI, ZBIGNIEW et al., "The Influence of Migration Sizes and Intervals on Island Models", June 29, 2005, 8 pages	
	9		Vahid Lari, et al., "Decentralized dynamic resource management support for massively parallel processor arrays", September 11, 2011, 8 pages	
	10		Vahid Lari, et al., "Distributed resource reservation in massively parallel processor arrays", May 16, 2011, 4 pages	
	11		Mohsen Hayati and Yazdan Shirvany, "Artificial Neural Network Approach for Short Term Load Forecasting for Illam Region", January 2007, 5 pages	
	12		Hasim Sak, Andrew Senior, and Francoise Beaufays, "Long Short-Term Memory Recurrent Neural Network Architectures for Large Scale Acoustic Modeling", January 2014, 5 pages	
	13		Jorg Walter and Klaus Schulten, "Implementation of self-organizing neural networks for visuo-motor control of an industrial robot", January 1993, 10 pages	
	14		Heikki Hyotyniemi and Aarno Lehtola, "A Universal Relation Database Interface for Knowledge Based Systems", April 1991, 5 pages	
	15		Mrissa, Michael, et al. "An avatar architecture for the web of things." IEEE Internet Computing 19.2 (2015): 30-38., 9 pages	
	16		Luck, Michael, and Ruth Aylett. "Applying artificial intelligence to virtual reality: Intelligent virtual environments." Applied Artificial Intelligence 14.1 (2000): 3-32., 30 pages	
	17		Terdjimi, Mehdi, et al. "An avatar-based adaptation workflow for the web of things." Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), 2016 IEEE 25th International Conference on. IEEE, 2016., 6 pages	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	18	Bogdanovych, Anton, et al. "Authentic interactive reenactment of cultural heritage with 3D virtual worlds and artificial intelligence." Applied Artificial Intelligence 24.6 (2010): 617-647., 32 pages	
	19	Hernandez, Marco E. Perez, and Stephan Reiff-Marganiec. "Autonomous and self controlling smart objects for the future internet." Future internet of things and cloud (FiCloud), 2015 3rd international conference on. IEEE, 2015., 8 pages	
	20	Medini et al., "Building a Web of Things with Avatars", Managing the Web of Things (2017), 30 pages	

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature	/SANTIAGO GARCIA/	Date Considered	06/19/2019

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Jasmin Cosic/	Date (YYYY-MM-DD)	2019-05-05
Name/Print	Jasmin Cosic	Registration Number	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

EAST Search History

EAST Search History (Prior Art)

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
L1	13297	G06N3/08.cpc.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/19 14:04
L2	6613	G06N5/022.cpc.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/19 14:04
L3	19658	1 or 2	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/19 14:04
L4	1287849	AI OR (((machine near2 learn$3) (artificial adj intelligence) (decision adj tree) (random adj forrest$1) (predictive adj model$4) (association adj rule) (neural adj network$3) (bayesian adj network$3)))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/19 14:04
L5	12487	3 and L4	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/19 14:04
L6	3171	L4 AND ((ANTICIPAT$4 OR PREDICT$5 or determ$5) NEAR4 (INSTRUCTION NEAR4 SETS))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/19 14:06
L7	105	5 and L6	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/19 14:06
L8	3215	L4 AND ((ANTICIPAT$4 OR PREDICT$5	US-PGPUB;	OR	ON	2019/06/19

		or determ$5) NEAR4 (INSTRUCTION NEAR4 SETS) or (current near4 compiler))	USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB			14:19
L9	0	((G06N3/08.CPC.)).CCLS.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2019/06/19 14:25
L12	6613	G06N5/022.CPC.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/19 14:26
L13	42	12 AND 8	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/19 14:26
L14	37	((("Cosic") near3 ("Jasmin"))).INV.	US-PGPUB; USPAT; USOCR	OR	ON	2019/06/19 14:39
S1	1	"15340991"	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/17 23:31
S2	990332	AI OR (ARTIFICIAL NEAR3 INTELLIGEN$4)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/17 23:32
S3	49	S2 AND (DEVICE NEAR4 CIRCUMSTANCES WITH AUTOMAT$5)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/17 23:33
S4	123	S2 AND ((ANTICIPAT$4 OR PREDICT$5) NEAR4 (INSTRUCTION NEAR4 SETS))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/17 23:43
S5	368	S2 AND ((ANTICIPAT$4 OR PREDICT$5) WITH (INSTRUCTION NEAR4 SETS))	US-PGPUB; USPAT; USOCR;	OR	ON	2019/06/17 23:43

			FPRS; EPO; JPO; DERWENT; IBM_TDB			
S6	94	S2 AND ((ANTICIPAT$4 OR PREDICT$5) WITH (INSTRUCTION NEAR4 SETS) WITH (OBJECT OR THING OR ITEM OR DEVICE))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/17 23:43
S7	1	S2 AND ((ANTICIPAT$4 OR PREDICT$5) WITH (training near4 INSTRUCTION NEAR4 SETS) WITH (OBJECT OR THING OR ITEM OR DEVICE))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/18 00:01
S8	0	S2 AND ((ANTICIPAT$4 OR PREDICT$5) WITH (training$ near4 INSTRUCTION NEAR4 SETS with train) WITH (OBJECT OR THING OR ITEM OR DEVICE))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/18 00:01
S9	280300	("20080254429" | "4730315" | "6973446" | "7840060" | "8196119" | "7113946" | "9298749" | "20100278420" | "20080215508" | "20090287643" | "20120284026" | "7565340" | "20130238533" | "6915105" | "7240335" | "20130226974" | "8335805" | "9282309" | "9443192" | "20080288259" | "20100033780" | "5560011" | "6842877" | "20040117771" | "20070006159" | "4860203" | "6088731" | "7424705" | "7484205" | "8090669" | "8251704" | "8397227" | "8464225" | "20050289105" | "20100023541" | "20100082536" | "20160246850" | "8572035" | "20040249774" | "20050245303" | "20150055821" | "20070061735" | "20110085734" | "20150264306" | "20160328480" | "20120150773" | "20120167057" | "5602982" | "6106299" | "6314558" | "7478371" | "7721218" | "7765537" | "8019699" | "8655260" | "8667472" | "20160140999" | "20160246868" | "20030026588" | "20070106633" | "20090131152" | "20090222388" | "20130278631" | "20050240412" | "20060265406" | "20100114746" | "20130278501" | "20150006171" | "20150310041" | "20150339213" | "6754631" | "4370707" | "6126330" | "6643842" | "6728689" | "8078556" | "8195674" | "8589414" | "20030065662" | "20130218932" | "9047324" | "9367806" | "20090141969" | "20100063949" |	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/18 00:52

		"20130156345" | "20140052717" | "20090067727" | "20130007532" | "8261199" | "20080281764" | "7797259" | "7987144" | "8005828" | "8244730" | "20040194017" | "20160292185" | "7117225" | "8417740" | "20060047612" | "20090324010" | "20060184410" | "20110007079" | "20150269415" | "8996432" | "9268454" | "6735632" | "7017153" | "7249349" | "8166463" | "8364612" | "8549359" | "20050149517" | "20050149542" | "20160142650" | "20160246819" | "20070058856" | "20080144893" | "20110030031" | "20150039304" | "20090044113" | "20090136095" | "20150324685" | "20160167226" | "9305216" | "20060190930" | "20100138370" | "20110218672" | "6026234" | "6801912" | "7052277" | "7222127" | "7925984" | "8137112" | "8266608" | "8386401" | "8655900" | "20040267521" | "20090110061" | "20100241595" | "20130159021" | "20140161250" | "20140177946" | "20140207580" | "20070050606" | "20070050719" | "20110270794" | "20120290347" | "20140075249" | "20140211988" | "20160274187")".PN"				
S10	13290	G06N3/08.cpc.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/18 10:57
S11	280300	("20080254429" | "4730315" | "6973446" | "7840060" | "8196119" | "7113946" | "9298749" | "20100278420" | "20080215508" | "20090287643" | "20120284026" | "7565340" | "20130238533" | "6915105" | "7240335" | "20130226974" | "8335805" | "9282309" | "9443192" | "20080288259" | "20100033780" | "5560011" | "6842877" | "20040117771" | "20070006159" | "4860203" | "6088731" | "7424705" | "7484205" | "8090669" | "8251704" | "8397227" | "8464225" | "20050289105" | "20100023541" | "20100082536" | "20160246850" | "8572035" | "20040249774" | "20050245303" | "20150055821" | "20070061735" | "20110085734" | "20150264306" | "20160328480" | "20120150773" | "20120167057" | "5602982" | "6106299" | "6314558" | "7478371" | "7721218" | "7765537" | "8019699" | "8655260" | "8667472" | "20160140999" | "20160246868" | "20030026588" | "20070106633" |	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/18 10:58

		"20090131152" | "20090222388" | "20130278631" | "20050240412" | "20060265406" | "20100114746" | "20130278501" | "20150006171" | "20150310041" | "20150339213" | "6754631" | "4370707" | "6126330" | "6643842" | "6728689" | "8078556" | "8195674" | "8589414" | "20030065662" | "20130218932" | "9047324" | "9367806" | "20090141969" | "20100063949" | "20130156345" | "20140052717" | "20090067727" | "20130007532" | "8261199" | "20080281764" | "7797259" | "7987144" | "8005828" | "8244730" | "20040194017" | "20160292185" | "7117225" | "8417740" | "20060047612" | "20090324010" | "20060184410" | "20110007079" | "20150269415" | "8996432" | "9268454" | "6735632" | "7017153" | "7249349" | "8166463" | "8364612" | "8549359" | "20050149517" | "20050149542" | "20160142650" | "20160246819" | "20070058856" | "20080144893" | "20110030031" | "20150039304" | "20090044113" | "20090136095" | "20150324685" | "20160167226" | "9305216" | "20060190930" | "20100138370" | "20110218672" | "6026234" | "6801912" | "7052277" | "7222127" | "7925984" | "8137112" | "8266608" | "8386401" | "8655900" | "20040267521" | "20090110061" | "20100241595" | "20130159021" | "20140161250" | "20140177946" | "20140207580" | "20070050606" | "20070050719" | "20110270794" | "20120290347" | "20140075249" | "20140211988" | "20160274187")".PN"				
S12	13290	G06N3/08.cpc.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/18 11:36
S13	280300	("20080254429" | "4730315" | "6973446" | "7840060" | "8196119" | "7113946" | "9298749" | "20100278420" | "20080215508" | "20090287643" | "20120284026" | "7565340" | "20130238533" | "6915105" | "7240335" | "20130226974" | "8335805" | "9282309" | "9443192" | "20080288259" | "20100033780" | "5560011" | "6842877" | "20040117771" | "20070006159" | "4860203" | "6088731" | "7424705" | "7484205" | "8090669" | "8251704" | "8397227" | "8464225" | "20050289105" | "20100023541" |	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/18 11:36

		"20100082536" | "20160246850" | "8572035" | "20040249774" | "20050245303" | "20150055821" | "20070061735" | "20110085734" | "20150264306" | "20160328480" | "20120150773" | "20120167057" | "5602982" | "6106299" | "6314558" | "7478371" | "7721218" | "7765537" | "8019699" | "8655260" | "8667472" | "20160140999" | "20160246868" | "20030026588" | "20070106633" | "20090131152" | "20090222388" | "20130278631" | "20050240412" | "20060265406" | "20100114746" | "20130278501" | "20150006171" | "20150310041" | "20150339213" | "6754631" | "4370707" | "6126330" | "6643842" | "6728689" | "8078556" | "8195674" | "8589414" | "20030065662" | "20130218932" | "9047324" | "9367806" | "20090141969" | "20100063949" | "20130156345" | "20140052717" | "20090067727" | "20130007532" | "8261199" | "20080281764" | "7797259" | "7987144" | "8005828" | "8244730" | "20040194017" | "20160292185" | "7117225" | "8417740" | "20060047612" | "20090324010" | "20060184410" | "20110007079" | "20150269415" | "8996432" | "9268454" | "6735632" | "7017153" | "7249349" | "8166463" | "8364612" | "8549359" | "20050149517" | "20050149542" | "20160142650" | "20160246819" | "20070058856" | "20080144893" | "20110030031" | "20150039304" | "20090044113" | "20090136095" | "20150324685" | "20160167226" | "9305216" | "20060190930" | "20100138370" | "20110218672" | "6026234" | "6801912" | "7052277" | "7222127" | "7925984" | "8137112" | "8266608" | "8386401" | "8655900" | "20040267521" | "20090110061" | "20100241595" | "20130159021" | "20140161250" | "20140177946" | "20140207580" | "20070050606" | "20070050719" | "20110270794" | "20120290347" | "20140075249" | "20140211988" | "20160274187")".PN"				
S14	78	S13 and S12	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/18 11:36
S15	990516	AI OR (ARTIFICIAL NEAR3 INTELLIGEN$4)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO;	OR	ON	2019/06/18 15:18

			DERWENT; IBM_TDB			
S16	36	S15 AND ((ANTICIPAT$4 OR PREDICT$5) WITH (INSTRUCTION NEAR4 SETS)) with correlat$4	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/18 15:18
S17	100	S15 AND ((ANTICIPAT$4 OR PREDICT$5) WITH (INSTRUCTION NEAR4 SETS)) with (compar$3 or match$3 or correlat$3 or associat$3 or relat$3)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/18 15:19
S18	387	("20080254429" | "4730315" | "6973446" | "7840060" | "8196119" | "7113946" | "9298749" | "20100278420" | "20080215508" | "20090287643" | "20120284026" | "7565340" | "20130238533" | "6915105" | "7240335" | "20130226974" | "8335805" | "9282309" | "9443192" | "20080288259" | "20100033780" | "5560011" | "6842877" | "20040117771" | "20070006159" | "4860203" | "6088731" | "7424705" | "7484205" | "8090669" | "8251704" | "8397227" | "8464225" | "20050289105" | "20100023541" | "20100082536" | "20160246850" | "8572035" | "20040249774" | "20050245303" | "20150055821" | "20070061735" | "20110085734" | "20150264306" | "20160328480" | "20120150773" | "20120167057" | "5602982" | "6106299" | "6314558" | "7478371" | "7721218" | "7765537" | "8019699" | "8655260" | "8667472" | "20160140999" | "20160246868" | "20030026588" | "20070106633" | "20090131152" | "20090222388" | "20130278631" | "20050240412" | "20060265406" | "20100114746" | "20130278501" | "20150006171" | "20150310041" | "20150339213" | "6754631" | "4370707" | "6126330" | "6643842" | "6728689" | "8078556" | "8195674" | "8589414" | "20030065662" | "20130218932" | "9047324" | "9367806" | "20090141969" | "20100063949" | "20130156345" | "20140052717" | "20090067727" | "20130007532" | "8261199" | "20080281764" | "7797259" | "7987144" | "8005828" | "8244730" | "20040194017" | "20160292185" | "7117225" | "8417740" | "20060047612" | "20090324010" | "20060184410" | "20110007079" | "20150269415" |	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/18 23:12

<table>
<tr><td></td><td></td><td>"8996432" | "9268454" | "6735632" | "7017153" | "7249349" | "8166463" | "8364612" | "8549359" | "20050149517" | "20050149542" | "20160142650" | "20160246819" | "20070058856" | "20080144893" | "20110030031" | "20150039304" | "20090044113" | "20090136095" | "20150324685" | "20160167226" | "9305216" | "20060190930" | "20100138370" | "20110218672" | "6026234" | "6801912" | "7052277" | "7222127" | "7925984" | "8137112" | "8266608" | "8386401" | "8655900" | "20040267521" | "20090110061" | "20100241595" | "20130159021" | "20140161250" | "20140177946" | "20140207580" | "20070050606" | "20070050719" | "20110270794" | "20120290347" | "20140075249" | "20140211988" | "20160274187").PN.</td><td></td><td></td><td></td><td></td></tr>
<tr><td>S19</td><td>133</td><td>("20010044732" | "20020078209" | "20020161735" | "20020174189" | "20030046451" | "20030065662" | "20040010699" | "20040049476" | "20040098394" | "20040128327" | "20040203845" | "20040224675" | "20040229595" | "20040249785" | "20040249857" | "20050044165" | "20050076069" | "20050086203" | "20050149517" | "20050149542" | "20050289105" | "20060136454" | "20060212846" | "20070049246" | "20070136264" | "20070291757" | "20080005054" | "20080039062" | "20080040782" | "20080045205" | "20080086481" | "20080091763" | "20080126403" | "20080172274" | "20080189250" | "20080281825" | "20090070149" | "20090106360" | "20090176529" | "20090201908" | "20090217375" | "20090254594" | "20090275331" | "20100023531" | "20100023541" | "20100077468" | "20100082536" | "20100088316" | "20100159903" | "20100223228" | "20100257376" | "20100278162" | "20110010344" | "20110016089" | "20110106448" | "20110119758" | "20110145210" | "20110145242" | "20110161290" | "20110258345" | "20110314482" | "20120016901" | "20120101993" | "20120131116" | "20120159393" | "20120173485" | "20120183221" | "20120209925" | "20120209948" | "20130036089" | "20130198171" | "4862390" | "5121470" | "5185857" | "5533029" | "5544222" | "5598534" | "5812117" | "5956637" | "6087952" | "6122527" | "6128012" | "6202060" | "6226665" | "6493717" | "6549625" | "6636873" | "6654762" | "6795706" | "6832084" | "6871068" | "6879989" | "6889139" |</td><td>US-PGPUB; USPAT; USOCR</td><td>OR</td><td>ON</td><td>2019/06/18 23:14</td></tr>
</table>

		"6895234" | "6959436" | "7113946" | "7117225" | "7328030" | "7444143" | "7450962" | "7483905").PN. OR ("7523112" | "7685127" | "7756525" | "7756829" | "7779029" | "7805397" | "7813714" | "7831554" | "7865924" | "7885635" | "7945687" | "7958270" | "8031694" | "8041372" | "8099332" | "8103272" | "8145663" | "8190645" | "8204911" | "8214409" | "8266187" | "8290898" | "8291408" | "8335805" | "8359223" | "8401539" | "8412150" | "8417740").PN. OR ("9367806").URPN.				
S20	1287849	AI OR (((machine near2 learn$3) (artificial adj intelligence) (decision adj tree) (random adj forrest$1) (predictive adj model$4) (association adj rule) (neural adj network$3) (bayesian adj network$3)))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/18 23:28
S21	711	S20 AND ((ANTICIPAT$4 OR PREDICT$5) NEAR4 (INSTRUCTION NEAR4 SETS))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/18 23:29
S22	3171	S20 AND ((ANTICIPAT$4 OR PREDICT$5 or determ$5) NEAR4 (INSTRUCTION NEAR4 SETS))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/18 23:32
S23	1	S20 AND ((ANTICIPAT$4 OR PREDICT$5 or determ$5) NEAR4 (INSTRUCTION NEAR4 SETS)) with (object near4 representation)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/18 23:32
S24	126	S20 AND ((ANTICIPAT$4 OR PREDICT$5 or determ$5) NEAR4 (INSTRUCTION NEAR4 SETS)) with (represent$6)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/18 23:33
S25	152	S20 AND ((ANTICIPAT$4 OR PREDICT$5 or determ$5) NEAR4 (INSTRUCTION NEAR4 SETS)) with (represent$6 or label$4)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/18 23:33
S26	3	((ANTICIPAT$4 OR PREDICT$5 or determ$5) NEAR4 (INSTRUCTION NEAR4 SETS)) with (represent$6 or label$4) same4 (third near4 correlat$4)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2019/06/19 09:17

EAST Search History (Interference)

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
L10	0	((G06N3/08.CPC.)).CCLS.	US-PGPUB; USPAT	OR	OFF	2019/06/19 14:25
L11	0	((G06N5/022.CPC.)).CCLS.	US-PGPUB; USPAT	OR	OFF	2019/06/19 14:26

6/19/2019 2:59:01 PM
C:\Users\sgarcia\Documents\EAST\Workspaces\15340991.wsp

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS

Examiner Initial*	Cite No	Patent Number	Kind Code1	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1					

If you wish to add additional U.S. Patent citation information please click the Add button.

U.S.PATENT APPLICATION PUBLICATIONS

Examiner Initial*	Cite No	Publication Number	Kind Code1	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	20030026588		2003-02-06	Elder, James H. ; et al.	
	2	20040249774		2004-12-09	Caid, William R. ; et al.	
	3	20040267521		2004-12-30	Cutler, Ross ; et al.	
	4	20050245303		2005-11-03	Graepel, Kurt Hartwig ; et al.	
	5	20060047612		2006-03-02	Stanley; Kenneth O. ; et al.	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	6	20070058856		2007-03-15	Boregowda; Lokesh R. ; et al.	
	7	20070106633		2007-05-10	Reiner; Bruce	
	8	20080144893		2008-06-19	GUO; Chun Biao ; et al.	
	9	20080288259		2008-11-20	Chambers; Robert L. ; et al.	
	10	20090110061		2009-04-30	LEE; Kang Yong ; et al.	
	11	20090131152		2009-05-21	Busse; Martin A.	
	12	20090141969		2009-06-04	Yu; Kai ; et al.	
	13	20090222388		2009-09-30	Hua; Wei ; et al.	
	14	20090324010		2009-12-31	Hou; Billy	
	15	20100033780		2010-02-11	Gitter; Kurt	
	16	20100063949		2010-03-11	EATON; JOHN ERIC ; et al.	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	17	20100241595		2010-09-23	Felsher; David Paul	
	18	20100278420		2010-11-04	Shet; Vinay Damodar ; et al.	
	19	20110030031		2011-02-03	Lussier; Paul ; et al.	
	20	20130156345		2013-06-20	Shmunk; Dmitry Valerievich	
	21	20130159021		2013-06-20	Felsher; David Paul	
	22	20130278631		2013-10-24	Border; John N. ; et al.	
	23	20140052717		2014-02-20	FRANKS; Kasian ; et al.	
	24	20140161250		2014-06-12	Leeds; Richard ; et al.	
	25	20140177946		2014-06-26	LIM; Kil-Taek ; et al.	
	26	20140207580		2014-06-24	Minnis; William Charles ; et al.	
	27	20150039304		2015-02-05	Wein; Ron	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	28	20150055821		2015-02-26	Fotland; David Allen	

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number[3]	Country Code[2] i	Kind Code[4]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T[5]
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T[5]
	1	ABOUT | OpenCV, retrieved from <URL: http://opencv.org/about.html> on Dec 13, 2014, 1 pages	
	2	Animetrics, Inc. | 3D Facial Recognition, retrieved from <URL: http://animetrics.com/> on Dec 13, 2014, 2 pages	
	3	Bag-of-words model, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	4	Beier–Neely morphing algorithm, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 1 pages	
	5	Computer vision, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 7 pages	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	
Application Number	15340991
Filing Date	2016-11-02
First Named Inventor	Jasmin Cosic
Art Unit	
Examiner Name	
Attorney Docket Number	

	6	Convolutional neural network, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 5 pages	
	7	Decimation (signal processing), retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 3 pages	
	8	Digital image processing, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 3 pages	
	9	Dissolve (filmmaking), retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 2 pages	
	10	Facial recognition system, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 6 pages	
	11	Feature detection (computer vision), retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 3 pages	
	12	Feature extraction, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	13	Gesture recognition, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 5 pages	
	14	GrabCut, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 1 pages	
	15	Image processing, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	16	Image segmentation, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 12 pages	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	17	Image warping, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 2 pages	
	18	Inbetweening, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 2 pages	
	19	Interpolation, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 5 pages	
	20	Language model, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 4 pages	
	21	List of speech recognition software, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 4 pages	
	22	Livewire Segmentation Technique, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 2 pages	
	23	Morphing, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 2 pages	
	24	Motion estimation, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	25	Muse: Face Morph || Mesh Warping, retrieved from <URL: http://alexwolfe.blogspot.com/2011/10/face-morph-mesh-warping.html> on Nov 19, 2015, 2 pages	
	26	n-gram, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 5 pages	
	27	Optical flow, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 4 pages	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	28	Outline of object recognition, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 7 pages	
	29	Phonotactics, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 3 pages	
	30	Recurrent neural network, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 5 pages	
	31	Sample rate conversion, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 2 pages	
	32	Simple interactive object extraction, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	33	Speech recognition, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 12 pages	
	34	Speech segmentation, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 4 pages	
	35	Welcome to recognize-speech.com, retrieved from <URL: http://recognize-speech.com/> on Oct 18, 2015, 1 pages	
	36	Introduction Speech, retrieved from <URL: http://recognize-speech.com/speech> on Oct 18, 2015, 1 pages	
	37	Preprocessing, retrieved from <URL: http://recognize-speech.com/preprocessing> on Oct 18, 2015, 4 pages	
	38	Feature Extraction, retrieved from <URL: http://recognize-speech.com/feature-extraction> on Oct 18, 2015, 3 pages	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	39	Acoustic model, retrieved from <URL: http://recognize-speech.com/acoustic-model> on Oct 18, 2015, 2 pages	
	40	Video content analysis, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	41	Video tracking, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 2 pages	
	42	Andrej Karpathy, Thomas Leung, George Toderici, Rahul Sukthankar, Sanketh Shetty, Li Fei-Fei, Large-scale Video Classification with Convolutional Neural Networks, Apr 14, 2014, 8 pages, Stanford University	
	43	Karen Simonyan, Andrew Zisserman, Two-Stream Convolutional Networks for Action Recognition in Videos, Nov 13, 2014, 11 pages, University of Oxford	

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature	/SANTIAGO GARCIA/	Date Considered	06/19/2019

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Jasmin Cosic/	Date (YYYY-MM-DD)	2016-11-10
Name/Print	Jasmin Cosic	Registration Number	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS Remove

Examiner Initial*	Cite No	Patent Number	Kind Code1	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	7113946		2006-09-26	Cosic	
	2	7117225		2006-10-03	Cosic	
	3	8335805		2012-12-18	Cosic	
	4	8417740		2013-04-09	Cosic	
	5	8572035		2013-10-29	Cosic	
	6	8655900		2014-02-18	Cosic	
	7	9047324		2015-06-02	Cosic	
	8	9282309		2016-03-08	Cosic	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number		15340991
	Filing Date		2016-11-02
	First Named Inventor	Jasmin Cosic	
	Art Unit		
	Examiner Name		
	Attorney Docket Number		

	9	9298749		2016-03-29	Cosic	
	10	9367806		2016-06-14	Cosic	
	11	9443192		2016-09-13	Cosic	

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS Remove

Examiner Initial*	Cite No	Publication Number	Kind Code1	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	20030065662		2003-04-03	Cosic	
	2	20040194017		2004-09-30	Cosic	
	3	20050149517		2005-07-07	Cosic	
	4	20050149542		2005-07-07	Cosic	
	5	20050289105		2005-12-29	Cosic	
	6	20100023541		2010-01-28	Cosic	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	7	20100082536		2010-04-01	Cosic	
	8	20130218932		2013-08-22	Cosic	
	9	20130226974		2013-08-29	Cosic	
	10	20160140999		2016-05-19	Cosic	
	11	20160142650		2016-05-19	Cosic	
	12	20160246819		2016-08-25	Cosic	
	13	20160246850		2016-08-25	Cosic	
	14	20160246868		2016-08-25	Cosic	
	15	20160292185		2016-10-06	Cosic	

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number3	Country Code2 i	Kind Code4	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T5

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T5
	1		

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature	/SANTIAGO GARCIA/	Date Considered	06/19/2019

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Jasmin Cosic/	Date (YYYY-MM-DD)	2016-11-10
Name/Print	Jasmin Cosic	Registration Number	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS Remove

Examiner Initial*	Cite No	Patent Number	Kind Code1	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	4370707		1983-01-25	Phillips , et al.	
	2	4730315		1988-03-08	Saito , et al.	
	3	4860203		1989-08-22	Corrigan , et al.	
	4	5602982		1997-02-11	Judd , et al.	
	5	6026234		2000-02-15	Hanson , et al.	
	6	6088731		2000-07-11	Kiraly , et al.	
	7	6106299		2000-08-22	Ackermann , et al.	
	8	6126330		2000-10-03	Knight	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	9	6314558		2001-11-06	Angel , et al.	
	10	6643842		2003-11-04	Angel , et al.	
	11	6728689		2004-04-27	Drissi , et al.	
	12	6735632		2004-05-11	Kiraly , et al.	
	13	6801912		2004-10-05	Moskowitz , et al.	
	14	6915105		2005-07-05	Masuda	
	15	6973446		2005-12-06	Mamitsuka , et al.	
	16	7017153		2006-03-21	Gouriou , et al.	
	17	7052277		2006-05-30	Kellman	
	18	7222127		2007-05-22	Bem , et al.	
	19	7240335		2007-07-03	Angel , et al.	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	
Application Number	15340991
Filing Date	2016-11-02
First Named Inventor	Jasmin Cosic
Art Unit	
Examiner Name	
Attorney Docket Number	

	20	7249349		2007-07-24	Hundt , et al.	
	21	7424705		2008-09-09	Lewis , et al.	
	22	7478371		2009-01-13	Gove	
	23	7484205		2009-01-27	Venkatapathy	
	24	7721218		2010-05-18	Awe , et al.	
	25	7765537		2010-07-27	Havin , et al.	
	26	7797259		2010-09-14	Jiang , et al.	
	27	7840060		2010-11-23	Podilchuk	
	28	7925984		2011-04-12	Awe , et al.	
	29	7987144		2011-07-26	Drissi , et al.	
	30	8005828		2011-08-23	Buchner , et al.	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	31	8019699		2011-09-13	Baxter	
	32	8078556		2011-12-13	Adi , et al.	
	33	8090669		2012-01-03	Shahani , et al.	
	34	8137112		2012-03-20	Woolf , et al.	
	35	8166463		2012-04-24	Gill , et al.	
	36	8195674		2012-06-05	Bem , et al.	
	37	8196119		2012-06-05	Gill , et al.	
	38	8244730		2012-08-14	Gupta	
	39	8251704		2012-08-28	Woolf , et al.	
	40	8266608		2012-09-11	Hecht , et al.	
	41	8364612		2013-01-29	Van Gael , et al.	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	42	8386401		2013-02-26	Virkar , et al.	
	43	8397227		2013-03-12	Fan , et al.	
	44	8464225		2013-06-11	Greifeneder	
	45	8549359		2013-10-01	Zheng	
	46	8589414		2013-11-19	Waite , et al.	
	47	8655260		2014-02-18	Yueh , et al.	
	48	8667472		2014-03-04	Molinari	

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS Remove

Examiner Initial*	Cite No	Publication Number	Kind Code1	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	20120167057		2012-06-28	Schmich; Christopher P. ; et al.	
	2	20120150773		2012-06-14	DiCorpo; Phillip ; et al.	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	3	20080254429		2008-10-16	Woolf; Susan D. ; et al.	
	4	20040117771		2004-06-17	Venkatapathy, Ramanathan	
	5	20130238533		2013-09-12	VIRKAR; Hemant ; et al.	
	6	20060190930		2006-08-24	Hecht; Daniel M. ; et al.	
	7	20070006159		2007-01-04	Hecht; Daniel Michael ; et al.	
	8	20080281764		2008-11-13	Baxter; Jonathan	
	9	20100138370		2010-06-03	Wu; Wang ; et al.	
	10	20110218672		2011-09-08	Maisonnier; Bruno ; et al.	

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number3	Country Code2 i	Kind Code4	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T5
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T5
	1	About Event Tracing, retrieved from <URL: http://msdn.microsoft.com/en-us/library/aa363668(d=default,l=en-us, v=vs.85).aspx> on Jan 12, 2014, 2 pages	
	2	Add dynamic Java code to your application, retrieved from <URL: http://www.javaworld.com/article/2071777/design-patterns/add-dynamic-java-code-to-your-application.html> on May 13, 2014, 6 pages	
	3	BCEL, retrieved from <URL: http://commons.apache.org> on May 13, 2014, 2 pages	
	4	Artificial intelligence, retrieved from <URL: http://wikipedia.com> on May 21, 2014, 28 pages	
	5	Artificial neural network, retrieved from <URL: http://wikipedia.com> on May 21, 2014, 12 pages	
	6	Branch (computer science), retrieved from <URL: http://wikipedia.com> on May 6, 2014, 2 pages	
	7	Branch trace, retrieved from <URL: http://wikipedia.com> on May 28, 2014, 2 pages	
	8	C Function Call Conventions and the Stack, retrieved from <URL: http://www.csee.umbc.edu/~chang/cs313.s02/stack.shtml> on Jan 11, 2014, 4 pages	
	9	Call stack, retrieved from <URL: http://wikipedia.com> on Jan 11, 2014, 5 pages	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	10	Program compilation and execution flow, retrieved from <URL: http://cs.stackexchange.com/questions/6187/program-compilation-and-execution-flow> on Jan 9, 2014, 2 pages	
	11	Control unit, retrieved from <URL: http://wikipedia.com> on May 25, 2014, 2 pages	
	12	Create dynamic applications with javax.tools, retrieved from <URL: http://www.ibm.com/developerworks/library/j-jcomp/> on May 13, 2014, 11 pages	
	13	Creating a Debugging and Profiling Agent with JVMTI, retrieved from <URL: http://www.oracle.com/technetwork/articles/java/jvmti-136367.html> on Jan 11, 2014, 18 pages	
	14	Decision tree learning, retrieved from <URL: http://wikipedia.com> on Jun 24, 2014, 5 pages	
	15	Dynamic Bytecode Instrumentation, retrieved from <URL: http://www.drdobbs.com/tools/dynamic-bytecode-instrumentation/184406433> on Apr 26, 2014, 6 pages	
	16	Dynamic programming language, retrieved from <URL: http://wikipedia.com> on May 7, 2014, 4 pages	
	17	Dynamic recompilation, retrieved from <URL: http://wikipedia.com> on May 19, 2014, 4 pages	
	18	eval(), retrieved from <URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/eval> on May 7, 2014, 13 pages	
	19	Function, retrieved from <URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Function> on May 7, 2014, 11 pages	
	20	FunctionEnter2 Function, retrieved from <URL: https://msdn.microsoft.com/en-us/library/aa964981(d=default,l=en-us,v=vs.110).aspx> on Mar 19, 2014, 2 pages	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number: 15340991
	Filing Date: 2016-11-02
	First Named Inventor: Jasmin Cosic
	Art Unit:
	Examiner Name:
	Attorney Docket Number:

	21	FunctionLeave2 Function, retrieved from <URL: https://msdn.microsoft.com/en-us/library/aa964942(d=default,l=en-us,v=vs.110).aspx> on Mar 19, 2014, 1 pages	
	22	Heuristic, retrieved from <URL: http://wikipedia.com> on May 22, 2014, 5 pages	
	23	How to: Configure Trace Switches, retrieved from <URL: https://msdn.microsoft.com/en-us/library/vstudio/t06xyy08(d=default,l=en-us,v=vs.100).aspx> on Mar 19, 2014, 3 pages	
	24	Indirect branch, retrieved from <URL: http://wikipedia.com> on May 5, 2014, 1 pages	
	25	Instruction cycle, retrieved from <URL: http://wikipedia.com> on Apr 30, 2014, 2 pages	
	26	Instruction set, retrieved from <URL: http://wikipedia.com> on Apr 29, 2014, 6 pages	
	27	Interrupt, retrieved from <URL: http://wikipedia.com> on May 15, 2014, 5 pages	
	28	Introduction to Instrumentation and Tracing, retrieved from <URL: http://msdn.microsoft.com/en-us/library/aa983649(d=default,l=en-us,v=vs.71).aspx> on Jan 10, 2014, 2 pages	
	29	Java bytecode, retrieved from <URL: http://wikipedia.com> on May 27, 2014, 4 pages	
	30	Package java.util.logging, retrieved from <URL: http://docs.oracle.com/javase/1.5.0/docs/api/java/util/logging/package-summary.html> on Mar 19, 2014, 2 pages	
	31	javac - Java programming language compiler, retrieved from <URL: http://docs.oracle.com/javase/7/docs/technotes/tools/windows/javac.html> on May 13, 2014, 12 pages	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	32	Interface JavaCompiler, retrieved from <URL: http://wikipedia.com> on May 13, 2014, 4 pages	
	33	Javassist, retrieved from <URL: http://wikipedia.com> on May 13, 2014, 2 pages	
	34	Just-in-time compilation, retrieved from <URL: http://wikipedia.com> on May 19, 2014, 5 pages	
	35	JVM Tool Interface, retrieved from <URL: http://docs.oracle.com/javase/7/docs/platform/jvmti/jvmti.html> on Apr 26, 2014, 127 pages	
	36	Machine code, retrieved from <URL: http://wikipedia.com> on Apr 30, 2014, 3 pages	
	37	Metaprogramming, retrieved from <URL: http://wikipedia.com> on May 8, 2014, 3 pages	
	38	Microsoft.VisualBasic.Logging Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/ms128122(d=default,l=en-us,v=vs.110).aspx> on Jan 13, 2014, 1 pages	
	39	Pin (computer program), retrieved from <URL: http://wikipedia.com> on May 6, 2014, 3 pages	
	40	Processor register, retrieved from <URL: http://wikipedia.com> on Apr 29, 2014, 4 pages	
	41	Profiling (computer programming), retrieved from <URL: http://wikipedia.com> on Jan 10, 2014, 4 pages	
	42	Profiling Overview, retrieved from <URL: http://msdn.microsoft.com/en-us/library/bb384493(d=default,l=en-us,v=vs.110).aspx> on Jan 12, 2014, 6 pages	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	43	Program counter, retrieved from <URL: http://wikipedia.com> on Apr 24, 2014, 2 pages	
	44	ptrace, retrieved from <URL: http://wikipedia.com> on Apr 24, 2014, 2 pages	
	45	Reflection (computer programming), retrieved from <URL: http://wikipedia.com> on May 7, 2014, 5 pages	
	46	Class Runtime, retrieved from <URL: http://docs.oracle.com/javase/7/docs/api/java/lang/Runtime.html> on Jan 10, 2014, 10 pages	
	47	Self-modifying code, retrieved from <URL: http://wikipedia.com> on Jan 12, 2014, 6 pages	
	48	Subjective logic, retrieved from <URL: http://wikipedia.com> on May 22, 2014, 5 pages	
	49	System.CodeDom.Compiler Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/z6b99ydt(d=default,l=en-us,v=vs.110).aspx> on May 13, 2014, 2 pages	
	50	System.Reflection.Emit Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/xd5fw18y(d=default,l=en-us,v=vs.110).aspx> on May 13, 2014, 2 pages	

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature	/SANTIAGO GARCIA/	Date Considered	06/19/2019

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Jasmin Cosic/	Date (YYYY-MM-DD)	2016-11-10
Name/Print	Jasmin Cosic	Registration Number	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS Remove

Examiner Initial*	Cite No	Patent Number	Kind Code1	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1					

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS Remove

Examiner Initial*	Cite No	Publication Number	Kind Code1	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1					

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number3	Country Code2 i	Kind Code4	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T5
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T5

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT **(Not for submission under 37 CFR 1.99)**	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	1	Tracing (software), retrieved from <URL: http://wikipedia.com> on Jan 10, 2014, 3 pages	
	2	Tree (data structure), retrieved from <URL: http://wikipedia.com> on Jun 24, 2014, 6 pages	
	3	PTRACE(2), retrieved from <URL: http://unixhelp.ed.ac.uk/CGI/man-cgi?ptrace> on Mar 19, 2014, 5 pages	
	4	Wevtutil, retrieved from <URL: http://technet.microsoft.com/en-us/library/cc732848(d=default,l=en-us,v=ws.11).aspx> on Apr 28, 2014, 5 pages	
	5	Intel Processor Trace, retrieved from <URL: https://software.intel.com/en-us/blogs/2013/09/18/processor-tracing> on Apr 28, 2014, 3 pages	
	6	YOUNGHOON JUNG, JAVA DYNAMICS Reflection and a lot more, Oct 10, 2012, 55 pages, Columbia University	
	7	AMITABH SRIVASTAVA, ALAN EUSTACE, ATOM A System for Building Customized Program Analysis Tools, May 3, 2004, 12 pages	
	8	MATHEW SMITHSON, KAPIL ANAND, APARNA KOTHA, KHALED ELWAZEER, NATHAN GILES, RAJEEV BARUA, Binary Rewriting without Relocation Information, Nov 10, 2010, 11 pages, University of Maryland	
	9	MAREK OLSZEWSKI, KEIR MIERTE, ADAM CZAJKOWSKI, ANGELA DEMLE BROWN, JIT Instrumentation - A Novel Approach To Dynamically Instrument Operating Systems, Feb 12, 2007, 14 pages, University of Toronto	

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature	/SANTIAGO GARCIA/	Date Considered	06/19/2019

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15340991
	Filing Date	2016-11-02
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Jasmin Cosic/	Date (YYYY-MM-DD)	2016-11-10
Name/Print	Jasmin Cosic	Registration Number	

This collection of information is required by 37 CFR 1.97 and 1.98. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 1 hour to complete, including gathering, preparing and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.G/

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Inventor:	Jasmin Cosic	Confirmation No.: 1993
Title:	"ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION"	
Serial No.:	15/340,991	Filed: November 2, 2016
Examiner:	GARCIA, SANTIAGO	Group Art Unit: 2668

Via EFS-Web July 18, 2019

COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, VA 22313-1450

COMMENTS ON STATEMENT OF REASONS FOR ALLOWANCE

Dear Commissioner:

In response to the Notice of Allowance (hereinafter "Notice of Allowance") dated June 24, 2019 concerning the above-identified application, the applicant respectfully submits the following Comments on Statement of Reasons for Allowance for the record.

A good summary of the claimed invention can be found in the abstract and independent claim 61.

With respect to at least some of the elements of the claimed invention that

prior art does not disclose, the Examiner notes that the claimed invention is allowable at least because “the prior art does not disclose the device that operates as the user directs it, hence, does not disclose learning the device operation from the user directing the device. Prior art discloses a system that explores its own environment on its own, which is very different from relying on the user to direct it. Prior art does not disclose the first and the second correlations that each include a circumstance representation correlated with instruction sets. Prior art does not disclose the knowledgebase, which includes a 3-level nested data structure recited in the independent claims. Such as individual circumstance representation (a data structure itself) and individual instruction set (a data structure itself when learned in the learning process). And correlation (a data structure itself) that includes a circumstance representation correlated with instruction sets. And knowledgebase a data structure itself including a plurality of correlation data structures in various arrangements. Further autonomous operation of the device in the claims performs the operation and does not rely on the re-training such as in the prior art as noted below, and the re-training is a crucial function in the proper operating of the systems in the prior art.” Each of the above-noted individual elements that prior art does not disclose and especially some or all of them combined render all claims of the claimed invention allowable.

With respect to at least some of the differences between the claimed invention and Schnittman, the training in Schnittman is different than the learning process recited in independent claims 61, 78, and 79 because the training in Schnittman does not include a user operating a device recited in independent claims 61, 78, and 79, and what is learned in the training in Schnittman is different than what is learned (i.e. at least a portion of a first and a second correlations each correlation including one or more circumstance representations correlated with one or more instruction sets, etc.) in the learning process recited in independent claims 61, 78, and 79. Therefore, the learning process itself is

different and the artificial knowledge learned in the learning process is different. Further, robot behavior or determining robot behavior in Schnittman is not the same as the “first one or more instruction sets for operating the first device learned in the learning process” recited in independent claims 61, 78, and 79 since robot behavior in Schnittman is not learned in a learning process that includes a user operating the robot. In fact, in Fig. 4 “Programmed Behaviors 430”, Schnittman teaches programmed robot behavior and, therefore, teaches away from instruction sets learned in a learning process that includes a user operating a device recited in independent claims 61, 78, and 79. Further, images, buffer, and database in Schnittman 545 are all different than a correlation including one or more circumstance representations correlated with one or more instruction sets. Further, the classifier in Schnittman is not the same as any element of independent claims 61, 78, and 79.

The above-noted include only a few of many claimed limitations that prior art does not disclose and other differences. In view of at least the above-noted claimed limitations that prior art does not disclose and other differences, the prior art references do not disclose the claimed invention, the specifics of the claimed invention, nor the broad spectrum idea.

The applicant respectfully submits that the allowed claims should not be construed as an admission that the claims as previously presented are not patentable, or that the cited references anticipate or render obvious the claims as previously presented. Indeed, the applicant respectfully submits that the claims as previously presented, and/or further amendments thereto, also contain patentable subject material. Thus, the applicant reserves the right to pursue the claims as previously presented, and/or further amendments thereto, in one or more continuation applications.

If the Examiner would like to discuss any aspect of this application, the

Examiner is respectfully requested to contact the undersigned at (317) 772-1312.

I hereby certify that this correspondence is being submitted electronically via EFS Web to the United States Patent and Trademark Office.

By /Jasmin Cosic/
Jasmin Cosic

Date submitted: July 18, 2019

Respectfully submitted,

/Jasmin Cosic/

Jasmin Cosic

Electronic Acknowledgement Receipt

EFS ID:	36629789
Application Number:	15340991
International Application Number:	
Confirmation Number:	1993
Title of Invention:	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION
First Named Inventor/Applicant Name:	Jasmin Cosic
Customer Number:	116094
Filer:	Jasmin Cosic
Filer Authorized By:	
Attorney Docket Number:	
Receipt Date:	18-JUL-2019
Filing Date:	02-NOV-2016
Time Stamp:	18:24:33
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	no

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1	Miscellaneous Incoming Letter	COMMENTS_ON_STATEMENT_OF_REASONS_FOR_ALLOWANCE.pdf	23561 7e822656e6102617ecc7c66e547b32ce057768ea	no	4

Warnings:

Information:	
Total Files Size (in bytes):	23561

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.
National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.
New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Inventor:	Jasmin Cosic	Confirmation No.: 1993
Title:	“ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION”	
Serial No.:	15/340,991	Filed: November 2, 2016
Examiner:	GARCIA, SANTIAGO	Group Art Unit: 2668

Via EFS-Web July 27, 2019

Mail Stop Amendment
COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, VA 22313-1450

AMENDMENT PURSUANT TO 37 C.F.R. 1.312

Dear Commissioner:

In response to notice of allowance (hereinafter “Notice of Allowance”) dated June 24, 2019 and pursuant to 37 C.F.R. 1.312, the applicant hereby amends the listing of claims that the Examiner filed in the Examiner's amendment in the Notice of Allowance.

Inventor: Jasmin Cosic
Serial No.: 15/340,991
Filing Date: November 2, 2016

Listing of Claims

61. (currently amended) A method implemented using a computing system that includes one or more processor circuits, the method comprising:

accessing a memory that stores at least a knowledgebase that includes: a first correlation including a first circumstance representation correlated with a first one or more instruction sets for operating a first device and a second correlation including a second circumstance representation correlated with a second one or more instruction sets for operating the first device, wherein the first circumstance representation represents a first circumstance detected at least in part by one or more sensors of the first device and the second circumstance representation represents a second circumstance detected at least in part by the one or more sensors of the first device, and wherein at least a portion of the first correlation and at least a portion of the second correlation are learned in a learning process that includes operating the first device at least partially by a user;

generating or receiving a third circumstance representation, wherein the third circumstance representation represents a third circumstance detected at least in part by the one or more sensors of the first device or at least in part by one or more sensors of a second device;

anticipating the first one or more instruction sets for operating the first device learned in the learning process based on at least partial match between the third circumstance representation and the first circumstance representation; and

at least in response to the anticipating, executing the first one or more instruction sets for operating the first device learned in the learning process, wherein the first device or ~~[[a]]~~ the second device autonomously performs one or more operations defined by the first one or more instruction sets for operating the first device learned in the learning process.

62. (previously presented) The method of claim 61, wherein the knowledgebase further includes: a first knowledge cell and a second knowledge cell, and wherein the first knowledge cell includes the first correlation and the second knowledge cell includes the second correlation.

63. (previously presented) The method of claim 61, wherein the learning process includes: creating, inserting, deleting, modifying, or manipulating an element of the first correlation, or creating, inserting, deleting, modifying, or manipulating an element of the second correlation.

64. (previously presented) The method of claim 61, wherein the learning process includes:

generating or receiving the first circumstance representation, and generating or receiving the second circumstance representation; and

obtaining or receiving the first one or more instruction sets for operating the first device, and obtaining or receiving the second one or more instruction sets for operating the first device, and wherein the knowledgebase further

includes a fourth correlation including a fourth circumstance representation correlated with a fourth one or more instruction sets for operating the second device, and wherein the fourth circumstance representation represents a fourth circumstance detected at least in part by the one or more sensors of the second device, and wherein at least a portion of the fourth correlation is learned in another learning process that includes:

generating or receiving the fourth circumstance representation; and

obtaining or receiving the fourth one or more instruction sets for operating the second device.

65. (currently amended) The method of claim 61, wherein the anticipating the first one or more instruction sets for operating the first device learned in the learning process based on the at least partial match between the third circumstance representation and the first circumstance representation includes:

determining that a number of at least partially matching portions of the third circumstance representation and portions of the first circumstance representation exceeds a threshold number[[;]], or

determining that a percentage of at least partially matching portions of the third circumstance representation and portions of the first circumstance representation exceeds a threshold percentage.

66. (previously presented) The method of claim 61, wherein the at least the portion of the first correlation and the at least the portion of the second

correlation are learned in the learning process while the user operates the first device, and wherein the one or more operations defined by the first one or more instruction sets for operating the first device learned in the learning process correspond to the user's methodology of operating the first device learned in the learning process.

67. (previously presented) The method of claim 61, wherein the knowledgebase further includes a fourth correlation including a fourth circumstance representation correlated with a fourth one or more instruction sets for operating: the first device, the second device, or a third device, and wherein a first connection is generated to connect the first correlation with the second correlation, and wherein a second connection is generated to connect the second correlation with the fourth correlation.

68. (previously presented) The method of claim 61, further comprising:

modifying the first one or more instruction sets for operating the first device learned in the learning process or a copy of the first one or more instruction sets for operating the first device learned in the learning process, and wherein the executing the first one or more instruction sets for operating the first device learned in the learning process includes executing the modified the first one or more instruction sets for operating the first device learned in the learning process or the modified the copy of the first one or more instruction sets for operating the first device learned in the learning process, and wherein the

performing, by the first device or by the second device, the one or more operations defined by the first one or more instruction sets for operating the first device learned in the learning process includes performing, by the first device or by the second device, one or more operations defined by the modified the first one or more instruction sets for operating the first device learned in the learning process or by the modified the copy of the first one or more instruction sets for operating the first device learned in the learning process.

69. (previously presented) The method of claim 61, wherein the knowledgebase further includes a fourth correlation including a fourth circumstance representation correlated with a fourth one or more instruction sets for operating the first device, and wherein the fourth circumstance representation represents a fourth circumstance detected at least in part by the one or more sensors of the first device, and wherein at least a portion of the fourth correlation is learned in another learning process that includes operating the first device at least partially by the user.

70. (previously presented) The method of claim 61, wherein the knowledgebase further includes a fourth correlation including a fourth circumstance representation correlated with a fourth one or more instruction sets for operating the first device, and wherein the fourth circumstance representation represents a fourth circumstance detected at least in part by the one or more sensors of the first device, and wherein at least a portion of the fourth correlation is learned in

another learning process that includes operating the first device at least partially by another user.

71. (previously presented) The method of claim 61, wherein the knowledgebase further includes a fourth correlation including a fourth circumstance representation correlated with a fourth one or more instruction sets for operating the second device, and wherein the fourth circumstance representation represents a fourth circumstance detected at least in part by the one or more sensors of the second device, and wherein at least a portion of the fourth correlation is learned in another learning process that includes operating the second device at least partially by another user.

72. (previously presented) The method of claim 61, wherein the first circumstance includes one or more objects detected at least in part by the one or more sensors of the first device at a first time or during a first time period, and wherein the second circumstance includes one or more objects detected at least in part by the one or more sensors of the first device at a second time or during a second time period, and wherein the third circumstance includes: one or more objects detected at least in part by the one or more sensors of the first device at a third time or during a third time period, or one or more objects detected at least in part by the one or more sensors of the second device at a third time or during a third time period.

73. (currently amended) The method of claim 61, wherein the first circumstance representation is a data structure including one or more data about the first circumstance of the first device, and wherein the second circumstance representation is a data structure including one or more data about the second circumstance of the first device, and wherein the third circumstance representation is a data structure including one or more data about: the third circumstance of the first device, or the third circumstance of the second device.

74. (previously presented) The method of claim 61, wherein the first circumstance representation includes: one or more object representations, or one or more collections of object representations, and wherein the second circumstance representation includes: one or more object representations, or one or more collections of object representations, and wherein the third circumstance representation includes: one or more object representations, or one or more collections of object representations.

75. (currently amended) The method of claim 61, wherein, to correlate the first circumstance representation with the first one or more instruction sets for operating the first device, a determination is made that the first one or more instruction sets for operating the first device temporally correspond to the first circumstance representation, and wherein the first circumstance representation includes: one or more object representations, or one or more collections of object representations.

76. (previously presented) The method of claim 61, wherein the anticipating the first one or more instruction sets for operating the first device learned in the learning process based on the at least partial match between the third circumstance representation and the first circumstance representation includes determining the first one or more instruction sets for operating the first device learned in the learning process based on the at least partial match between the third circumstance representation and the first circumstance representation.

77. (previously presented) The method of claim 61, wherein elements of the computing system are included in: a single device, or multiple devices, and wherein the one or more processor circuits include: one or more microcontrollers, one or more computing circuits, or one or more electronic circuits, and wherein the memory includes: a volatile memory, or a non-volatile memory, and wherein the first device includes: a robot, a vehicle, an appliance, an electronic device, or a mechanical machine, and wherein the second device includes: a robot, a vehicle, an appliance, an electronic device, or a mechanical machine, and wherein an instruction set of the first one or more instruction sets for operating the first device includes at least one selected from the group comprising: only one instruction, a plurality of instructions, one or more inputs, one or more commands, one or more computer commands, one or more keywords, one or more symbols, one or more operators, one or more variables, one or more values, one or more objects or object references, one or more data structures or

data structure references, one or more functions or function references, one or more parameters, one or more signals, one or more characters, one or more digits, one or more user operating directions, one or more user directions, one or more user inputs, one or more representations of one or more user actions, one or more representations of one or more user clicks, one or more binary bits, one or more assembly language commands, one or more states, one or more codes, one or more data, and one or more information, and wherein an instruction set of the second one or more instruction sets for operating the first device includes at least one selected from the group comprising: only one instruction, a plurality of instructions, one or more inputs, one or more commands, one or more computer commands, one or more keywords, one or more symbols, one or more operators, one or more variables, one or more values, one or more objects or object references, one or more data structures or data structure references, one or more functions or function references, one or more parameters, one or more signals, one or more characters, one or more digits, one or more user operating directions, one or more user directions, one or more user inputs, one or more representations of one or more user actions, one or more representations of one or more user clicks, one or more binary bits, one or more assembly language commands, one or more states, one or more codes, one or more data, and one or more information, and wherein the one or more sensors of the first device include at least one selected from the group comprising: one or more cameras, one or more microphones, one or more radars, one or more lidars, one or more sonars, and one or more apparatuses for detecting objects or object properties,

and wherein the one or more sensors of the second device include at least one selected from the group comprising: one or more cameras, one or more microphones, one or more radars, one or more lidars, one or more sonars, and one or more apparatuses for detecting objects or object properties, and wherein the at least the portion of the first correlation includes: one portion of the first correlation, multiple portions of the first correlation, all portions of the first correlation, or the entire first correlation, and wherein the at least the portion of the second correlation includes: one portion of the second correlation, multiple portions of the second correlation, all portions of the second correlation, or the entire second correlation, and wherein an object of the first circumstance is the same as an object of the third circumstance, or multiple objects of the first circumstance are the same as multiple objects of the third circumstance, or all objects of the first circumstance are the same as all objects of the third circumstance, or all objects of the first circumstance are different than all objects of the third circumstance.

78. (currently amended) One or more non-transitory machine readable media storing machine readable code that when executed by one or more processor circuits causes the one or more processor circuits to perform at least:

accessing a memory that stores at least a knowledgebase that includes: a first correlation including a first circumstance representation correlated with a first one or more instruction sets for operating a first device and a second correlation including a second circumstance representation correlated with a second one or

more instruction sets for operating the first device, wherein the first circumstance representation represents a first circumstance detected at least in part by one or more sensors of the first device and the second circumstance representation represents a second circumstance detected at least in part by the one or more sensors of the first device, and wherein at least a portion of the first correlation and at least a portion of the second correlation are learned in a learning process that includes operating the first device at least partially by a user;

generating or receiving a third circumstance representation, wherein the third circumstance representation represents a third circumstance detected at least in part by the one or more sensors of the first device or at least in part by one or more sensors of a second device;

anticipating the first one or more instruction sets for operating the first device learned in the learning process based on at least partial match between the third circumstance representation and the first circumstance representation; and

at least in response to the anticipating, causing the first device or ~~[[a]]~~ the second device to perform one or more operations defined by the first one or more instruction sets for operating the first device learned in the learning process at least by causing the one or more processor circuits or another one or more processor circuits to execute the first one or more instruction sets for operating the first device learned in the learning process.

79. (currently amended) A system comprising:

a memory that stores at least a knowledgebase that includes: a first correlation including a first circumstance representation correlated with a first one or more instruction sets for operating a first device and a second correlation including a second circumstance representation correlated with a second one or more instruction sets for operating the first device, wherein the first circumstance representation represents a first circumstance detected at least in part by one or more sensors of the first device and the second circumstance representation represents a second circumstance detected at least in part by the one or more sensors of the first device, and wherein at least a portion of the first correlation and at least a portion of the second correlation are learned in a learning process that includes operating the first device at least partially by a user;

means for generating or receiving a third circumstance representation, wherein the third circumstance representation represents a third circumstance detected at least in part by the one or more sensors of the first device or at least in part by one or more sensors of a second device;

means for anticipating the first one or more instruction sets for operating the first device learned in the learning process based on at least partial match between the third circumstance representation and the first circumstance representation; and

means for executing, at least in response to the anticipating, the first one or more instruction sets for operating the first device learned in the learning process, wherein the first device or ~~[[a]]~~ the second device autonomously

performs one or more operations defined by the first one or more instruction sets for operating the first device learned in the learning process.

80. (previously presented) The system of claim 79, wherein the means for generating or receiving the third circumstance representation includes one or more processor circuits, and wherein the means for anticipating the first one or more instruction sets for operating the first device learned in the learning process based on the at least partial match between the third circumstance representation and the first circumstance representation includes one or more processor circuits, and wherein the means for executing, at least in response to the anticipating, the first one or more instruction sets for operating the first device learned in the learning process includes one or more processor circuits.

Remarks

I. 37 C.F.R. 1.312 Amendment

Entry of the foregoing amendments is respectfully requested pursuant to 37 C.F.R. 1.312.

The amendments to claims 61, 65, 73, 75, 78, and 79 are needed to correct minor typographical errors as indicated.

No additional search or examination is needed because the claims correct minor typographical errors. The claims are patentable for the same reasons as those outlined in the Notice of Allowance.

If the Examiner would like to discuss any aspect of this application, the Examiner is respectfully requested to contact the undersigned at (317) 772-1312.

I hereby certify that this correspondence is being submitted electronically via EFS Web to the United States Patent and Trademark Office.

By /Jasmin Cosic/
Jasmin Cosic

Date submitted: July 27, 2019

Respectfully submitted,

/Jasmin Cosic/

Jasmin Cosic

Electronic Acknowledgement Receipt

EFS ID:	36707654
Application Number:	15340991
International Application Number:	
Confirmation Number:	1993
Title of Invention:	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION
First Named Inventor/Applicant Name:	Jasmin Cosic
Customer Number:	116094
Filer:	Jasmin Cosic
Filer Authorized By:	
Attorney Docket Number:	
Receipt Date:	27-JUL-2019
Filing Date:	02-NOV-2016
Time Stamp:	21:34:14
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	no

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1		312_Amendment.pdf	57078 a154a65003ae6c42ae7b7d16122246d6ccd2ff4b	yes	15

Multipart Description/PDF files in .zip description		
Document Description	**Start**	**End**
Amendment after Notice of Allowance (Rule 312)	1	1
Claims	2	14
Applicant Arguments/Remarks Made in an Amendment	15	15

Warnings:

Information:

Total Files Size (in bytes):	57078

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.

National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
15/340,991	11/02/2016	Jasmin Cosic		1993

116094 7590 08/08/2019
Jasmin Cosic
108 Woodbury Street
Pawtucket, RI 02861

EXAMINER	
GARCIA, SANTIAGO	

ART UNIT	PAPER NUMBER
2668	

MAIL DATE	DELIVERY MODE
08/08/2019	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Response to Rule 312 Communication	**Application No.** 15/340,991	**Applicant(s)** Cosic, Jasmin	
	Examiner SANTIAGO GARCIA	**Art Unit** 2668	**AIA Status** Yes

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

1. ☑ The amendment filed on 27 July 2019 under 37 CFR 1.312 has been considered, and has been:
   - a) ☐ entered.
   - b) ☑ entered as directed to matters of form not affecting the scope of the invention.
   - c) ☐ disapproved because the amendment was filed after the payment of the issue fee. Any amendment filed after the date the issue fee is paid must be accompanied by a petition under 37 CFR 1.313(c)(1) and the required fee to withdraw the application from issue.
   - d) ☐ disapproved. See explanation below.
   - e) ☐ entered in part. See explanation below.

/SANTIAGO GARCIA/
Primary Examiner, Art Unit 2668

PART B - FEE(S) TRANSMITTAL

Complete and send this form, together with applicable fee(s), by mail or fax, or via EFS-Web.

By mail, send to: Mail Stop ISSUE FEE
Commissioner for Patents
P.O. Box 1450
Alexandria, Virginia 22313-1450

By fax, send to: (571)-273-2885

INSTRUCTIONS: This form should be used for transmitting the ISSUE FEE and PUBLICATION FEE (if required). Blocks 1 through 5 should be completed where appropriate. All further correspondence including the Patent, advance orders and notification of maintenance fees will be mailed to the current correspondence address as indicated unless corrected below or directed otherwise in Block 1, by (a) specifying a new correspondence address; and/or (b) indicating a separate "FEE ADDRESS" for maintenance fee notifications.

CURRENT CORRESPONDENCE ADDRESS (Note: Use Block 1 for any change of address)

116094 7590 06/24/2019

Jasmin Cosic
108 Woodbury Street
Pawtucket, RI 02861

Note: A certificate of mailing can only be used for domestic mailings of the Fee(s) Transmittal. This certificate cannot be used for any other accompanying papers. Each additional paper, such as an assignment or formal drawing, must have its own certificate of mailing or transmission.

Certificate of Mailing or Transmission

I hereby certify that this Fee(s) Transmittal is being deposited with the United States Postal Service with sufficient postage for first class mail in an envelope addressed to the Mail Stop ISSUE FEE address above, or being transmitted to the USPTO via EFS-Web or by facsimile to (571) 273-2885, on the date below.

Jasmin Cosic (Typed or printed name)
/Jasmin Cosic/ (Signature)
August 16, 2019 (Date)

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
15/340,991	11/02/2016	Jasmin Cosic		1993

TITLE OF INVENTION: ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION

APPLN. TYPE	ENTITY STATUS	ISSUE FEE DUE	PUBLICATION FEE DUE	PREV. PAID ISSUE FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	SMALL	$500	$0.00	$0.00	$500	09/24/2019

EXAMINER	ART UNIT	CLASS-SUBCLASS
GARCIA, SANTIAGO	2668	382-155000

1. Change of correspondence address or indication of "Fee Address" (37 CFR 1.363).

☐ Change of correspondence address (or Change of Correspondence Address form PTO/SB/122) attached.

☐ "Fee Address" indication (or "Fee Address" Indication form PTO/SB/47; Rev 03-09 or more recent) attached. **Use of a Customer Number is required.**

2. For printing on the patent front page, list
(1) The names of up to 3 registered patent attorneys or agents OR, alternatively,
(2) The name of a single firm (having as a member a registered attorney or agent) and the names of up to 2 registered patent attorneys or agents. If no name is listed, no name will be printed.

1 ____
2 ____
3 ____

3. ASSIGNEE NAME AND RESIDENCE DATA TO BE PRINTED ON THE PATENT (print or type)

PLEASE NOTE: Unless an assignee is identified below, no assignee data will appear on the patent. If an assignee is identified below, the document must have been previously recorded, or filed for recordation, as set forth in 37 CFR 3.11 and 37 CFR 3.81(a). Completion of this form is NOT a substitute for filing an assignment.

(A) NAME OF ASSIGNEE

(B) RESIDENCE: (CITY and STATE OR COUNTRY)

Please check the appropriate assignee category or categories (will not be printed on the patent) : ☐ Individual ☐ Corporation or other private group entity ☐ Government

4a. Fees submitted: ☒ Issue Fee ☐ Publication Fee (if required) ☐ Advance Order - # of Copies ____

4b. Method of Payment: *(Please first reapply any previously paid fee shown above)*

☒ Electronic Payment via EFS-Web ☐ Enclosed check ☐ Non-electronic payment by credit card (Attach form PTO-2038)

☐ The Director is hereby authorized to charge the required fee(s), any deficiency, or credit any overpayment to Deposit Account No. ____

5. **Change in Entity Status** (from status indicated above)

☐ Applicant certifying micro entity status. See 37 CFR 1.29

☐ Applicant asserting small entity status. See 37 CFR 1.27

☐ Applicant changing to regular undiscounted fee status.

NOTE: Absent a valid certification of Micro Entity Status (see forms PTO/SB/15A and 15B), issue fee payment in the micro entity amount will not be accepted at the risk of application abandonment.
NOTE: If the application was previously under micro entity status, checking this box will be taken to be a notification of loss of entitlement to micro entity status.
NOTE: Checking this box will be taken to be a notification of loss of entitlement to small or micro entity status, as applicable.

NOTE: This form must be signed in accordance with 37 CFR 1.31 and 1.33. See 37 CFR 1.4 for signature requirements and certifications.

Authorized Signature /Jasmin Cosic/ Date August 16, 2019

Typed or printed name Jasmin Cosic Registration No. ____

Electronic Patent Application Fee Transmittal

Application Number:	15340991
Filing Date:	02-Nov-2016
Title of Invention:	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION
First Named Inventor/Applicant Name:	Jasmin Cosic
Filer:	Jasmin Cosic
Attorney Docket Number:	

Filed as Small Entity

Filing Fees for **Utility under 35 USC 111(a)**

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Basic Filing:				
Pages:				
Claims:				
Miscellaneous-Filing:				
Petition:				
Patent-Appeals-and-Interference:				
Post-Allowance-and-Post-Issuance:				
UTILITY APPL ISSUE FEE	2501	1	500	500

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Extension-of-Time:				
Miscellaneous:				
		Total in USD ($)		**500**

Electronic Acknowledgement Receipt

EFS ID:	36903735
Application Number:	15340991
International Application Number:	
Confirmation Number:	1993
Title of Invention:	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION
First Named Inventor/Applicant Name:	Jasmin Cosic
Customer Number:	116094
Filer:	Jasmin Cosic
Filer Authorized By:	
Attorney Docket Number:	
Receipt Date:	16-AUG-2019
Filing Date:	02-NOV-2016
Time Stamp:	18:14:30
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	yes
Payment Type	CARD
Payment was successfully received in RAM	$500
RAM confirmation Number	E20198FI18163567
Deposit Account	
Authorized User	

The Director of the USPTO is hereby authorized to charge indicated fees and credit any overpayment as follows:

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1	Issue Fee Payment (PTO-85B)	PTOL-85B_Issue_Fee_Form.pdf	87585 25f61878f8d47c7d0023a3252e8d625208f8230f	no	1
Warnings:					
Information:					
2	Fee Worksheet (SB06)	fee-info.pdf	29903 a1443dbe45e9b7975c9904ab21d1f5d6b0023dd8	no	2
Warnings:					
Information:					
			Total Files Size (in bytes):	117488	

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

<u>New Applications Under 35 U.S.C. 111</u>
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.
<u>National Stage of an International Application under 35 U.S.C. 371</u>
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.
<u>New International Application Filed with the USPTO as a Receiving Office</u>
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	ISSUE DATE	PATENT NO.	ATTORNEY DOCKET NO.	CONFIRMATION NO.
15/340,991	10/22/2019	10452974		1993

116094 7590 10/02/2019

Jasmin Cosic
108 Woodbury Street
Pawtucket, RI 02861

ISSUE NOTIFICATION

The projected patent number and issue date are specified above.

Determination of Patent Term Adjustment under 35 U.S.C. 154 (b)
(application filed on or after May 29, 2000)

The Patent Term Adjustment is 525 day(s). Any patent to issue from the above-identified application will include an indication of the adjustment on the front page.

If a Continued Prosecution Application (CPA) was filed in the above-identified application, the filing date that determines Patent Term Adjustment is the filing date of the most recent CPA.

Applicant will be able to obtain more detailed information by accessing the Patent Application Information Retrieval (PAIR) WEB site (http://pair.uspto.gov).

Any questions regarding the Patent Term Extension or Adjustment determination should be directed to the Office of Patent Legal Administration at (571)-272-7702. Questions relating to issue and publication fee payments should be directed to the Application Assistance Unit (AAU) of the Office of Data Management (ODM) at (571)-272-4200.

APPLICANT(s) (Please see PAIR WEB site http://pair.uspto.gov for additional applicants):

Jasmin Cosic, Miami, FL;

Exhibit N

DocCode – SCORE

SCORE Placeholder Sheet for IFW Content

Application Number: 16584736 Document Date: 09/26/2019

The presence of this form in the IFW record indicates that the following document type was received in electronic format on the date identified above. This content is stored in the SCORE database.

Since this was an electronic submission, there is no physical artifact folder, no artifact folder is recorded in PALM, and no paper documents or physical media exist. The TIFF images in the IFW record were created from the original documents that are stored in SCORE.

- Drawing

At the time of document entry (noted above):

- USPTO employees may access SCORE content via DAV or via the SCORE web page.
- External customers may access SCORE content via PAIR using the Supplemental Content tab.

ABSTRACT

Aspects of the disclosure generally relate to computing enabled systems and/or devices and may be generally directed to machine learning for computing enabled systems and/or devices. In some aspects, the system captures one or more digital pictures, receives one or more instruction sets, and learns correlations between the captured pictures and the received instruction sets.

PTO/AIA/01 (06-12)
Approved for use through 01/31/2014. OMB 0651-0032
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

DECLARATION (37 CFR 1.63) FOR UTILITY OR DESIGN APPLICATION USING AN APPLICATION DATA SHEET (37 CFR 1.76)

Title of Invention	MACHINE LEARNING FOR COMPUTING ENABLED SYSTEMS AND/OR DEVICES

As the below named inventor, I hereby declare that:

This declaration is directed to:

- [x] The attached application, or
- [ ] United States application or PCT international application number ____________________

filed on ______________________.

The above-identified application was made or authorized to be made by me.

I believe that I am the original inventor or an original joint inventor of a claimed invention in the application.

I hereby acknowledge that any willful false statement made in this declaration is punishable under 18 U.S.C. 1001 by fine or imprisonment of not more than five (5) years, or both.

WARNING:

Petitioner/applicant is cautioned to avoid submitting personal information in documents filed in a patent application that may contribute to identity theft. Personal information such as social security numbers, bank account numbers, or credit card numbers (other than a check or credit card authorization form PTO-2038 submitted for payment purposes) is never required by the USPTO to support a petition or an application. If this type of personal information is included in documents submitted to the USPTO, petitioners/applicants should consider redacting such personal information from the documents before submitting them to the USPTO. Petitioner/applicant is advised that the record of a patent application is available to the public after publication of the application (unless a non-publication request in compliance with 37 CFR 1.213(a) is made in the application) or issuance of a patent. Furthermore, the record from an abandoned application may also be available to the public if the application is referenced in a published application or an issued patent (see 37 CFR 1.14). Checks and credit card authorization forms PTO-2038 submitted for payment purposes are not retained in the application file and therefore are not publicly available.

LEGAL NAME OF INVENTOR

Inventor: Jasmin Cosic Date (Optional) : ______________

Signature: ______________________

Note: An application data sheet (PTO/SB/14 or equivalent), including naming the entire inventive entity, must accompany this form or must have been previously filed. Use an additional PTO/AIA/01 form for each additional inventor.

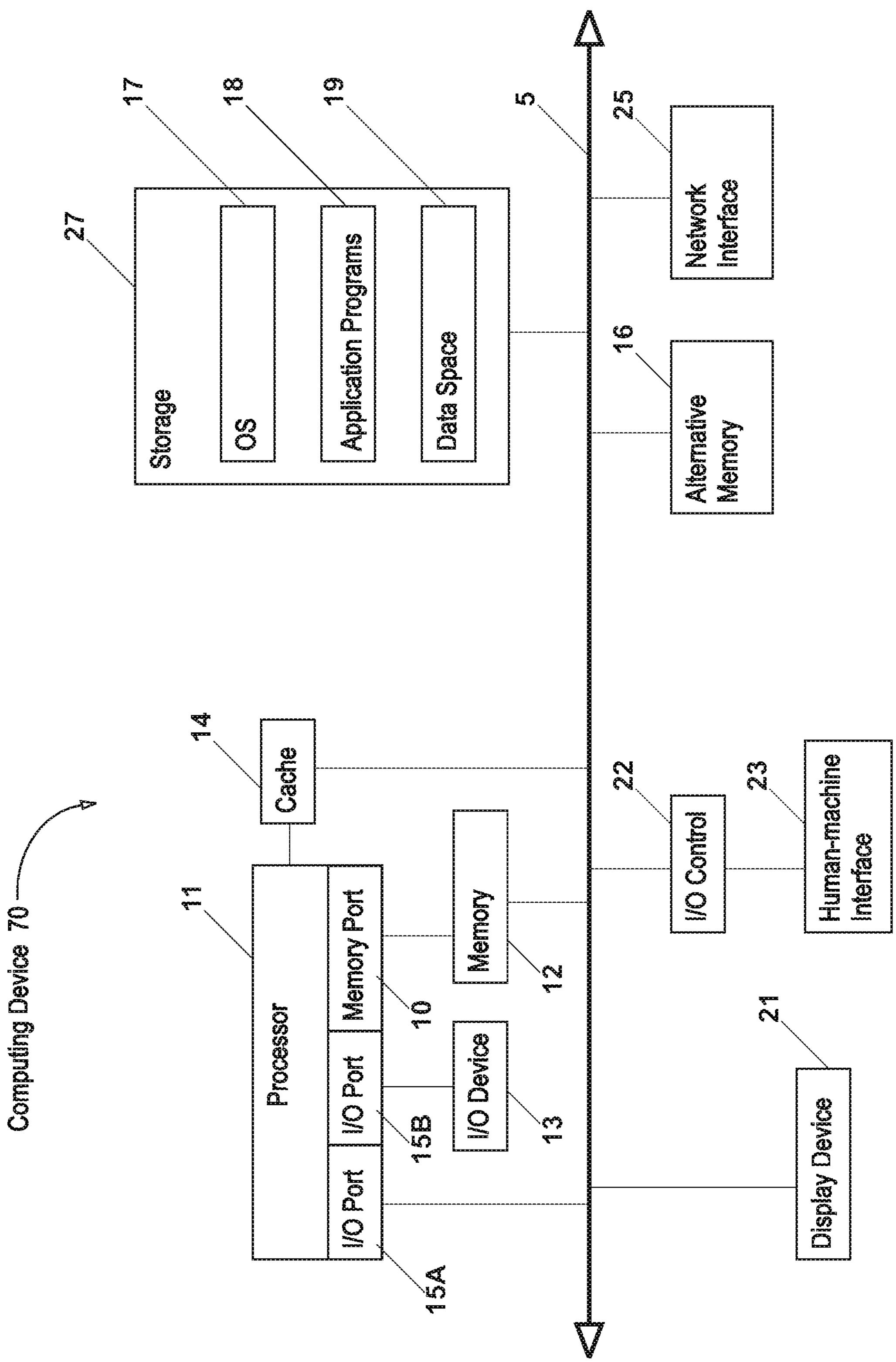

Computing Device 70
Processor
11
I/O Port
15A
I/O Port
15B
Memory Port
10
Cache
14
I/O Device
13
Memory
12
Storage
27
OS
17
Application Programs
18
Data Space
19
5
Network Interface
25
Alternative Memory
16
I/O Control
22
Human-machine Interface
23
Display Device
21

FIG. 1

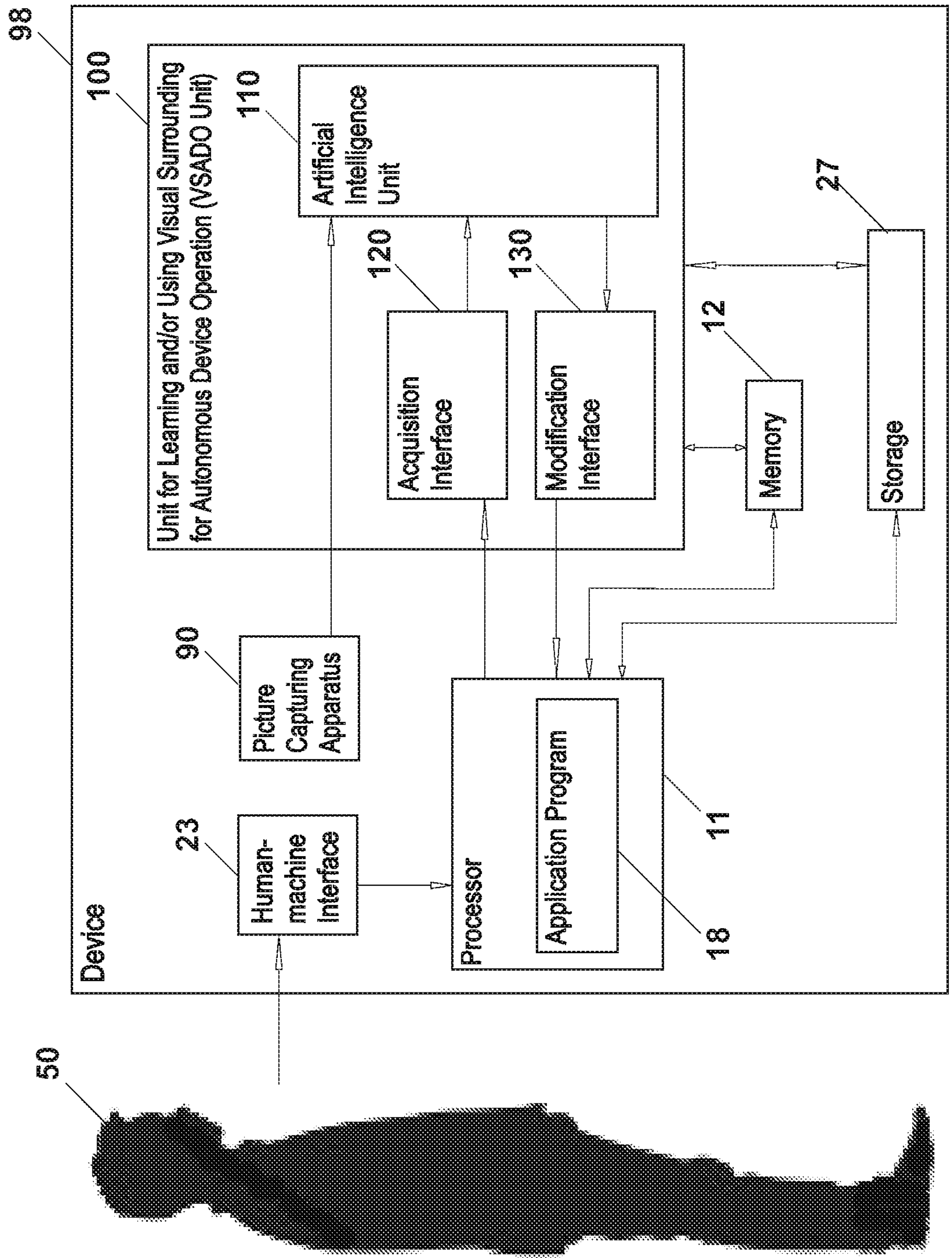

98
Device
100
Unit for Learning and/or Using Visual Surrounding for Autonomous Device Operation (VSADO Unit)
110
Artificial Intelligence Unit
120
Acquisition Interface
130
Modification Interface
12
Memory
27
Storage
90
Picture Capturing Apparatus
23
Human-machine Interface
Processor
Application Program
11
18
50

FIG. 2

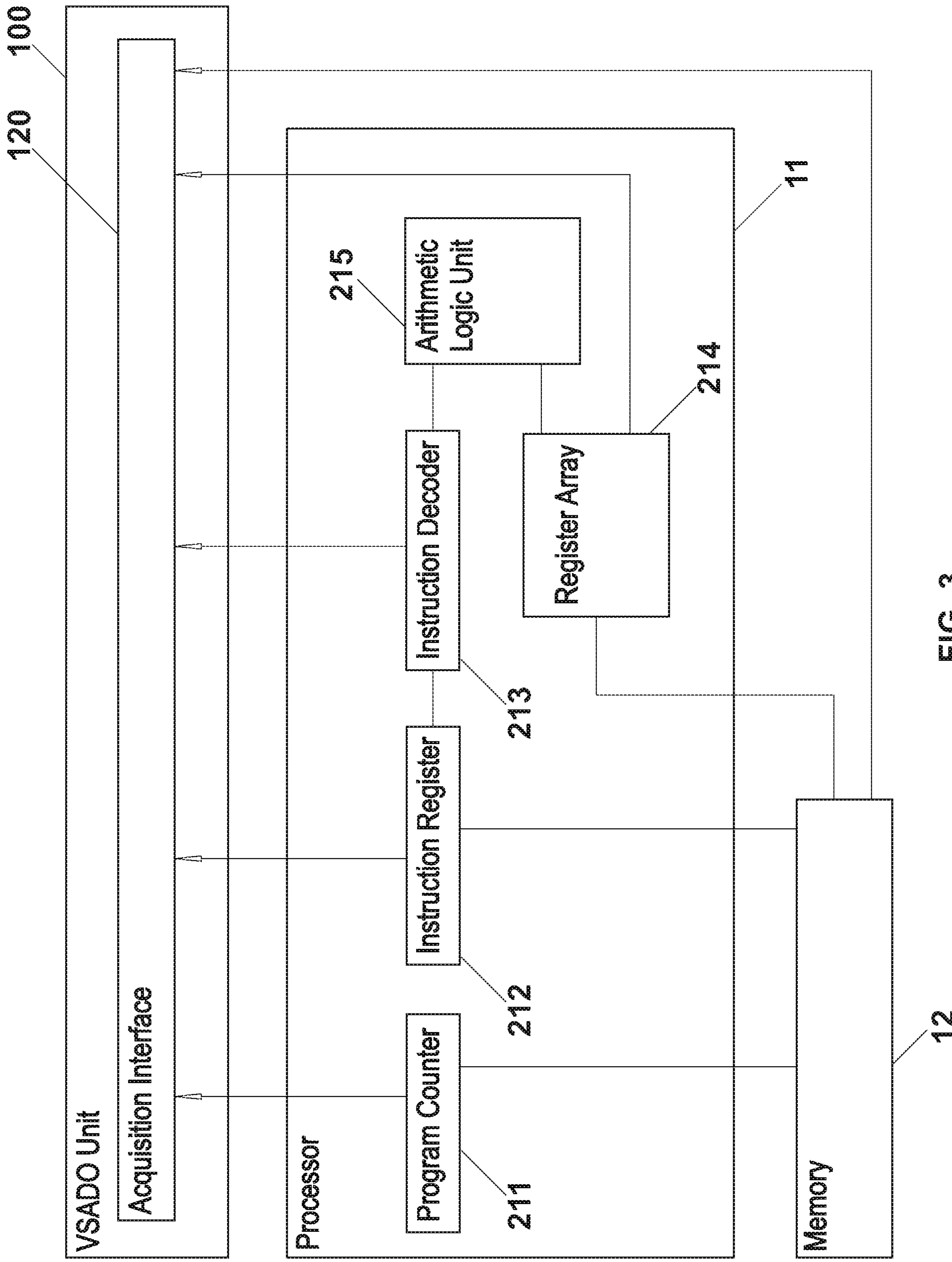

100
120
VSADO Unit
Acquisition Interface
Processor
11
215
Arithmetic Logic Unit
214
Instruction Decoder
Register Array
213
Instruction Register
212
Program Counter
211
Memory
12

FIG. 3

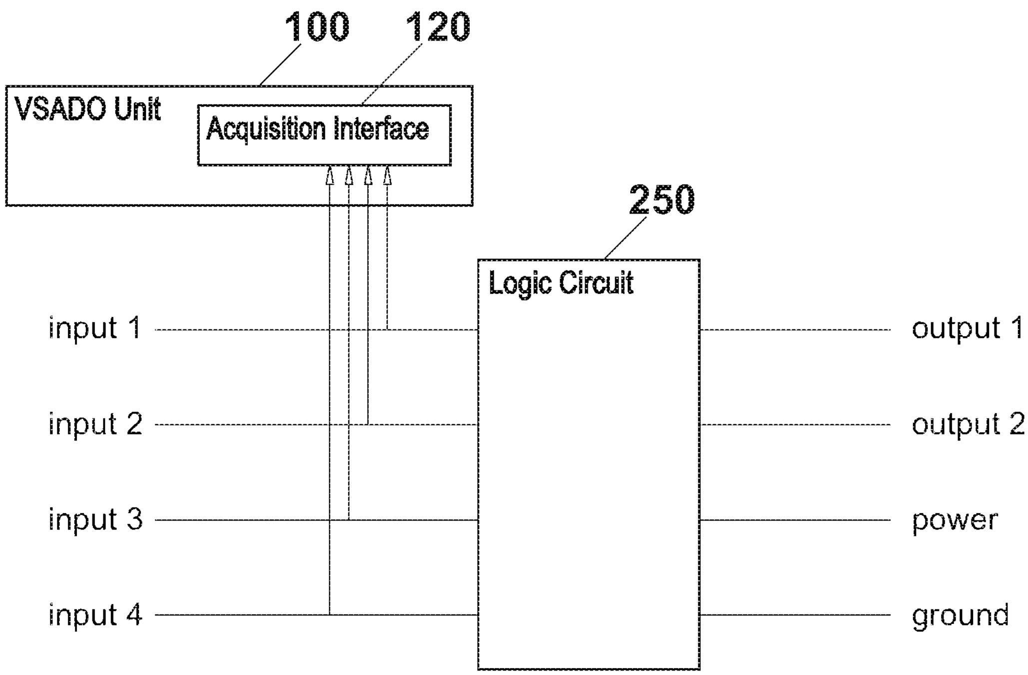

100
120
VSADO Unit
Acquisition Interface
250
Logic Circuit
input 1
input 2
input 3
input 4
output 1
output 2
power
ground

FIG. 4A

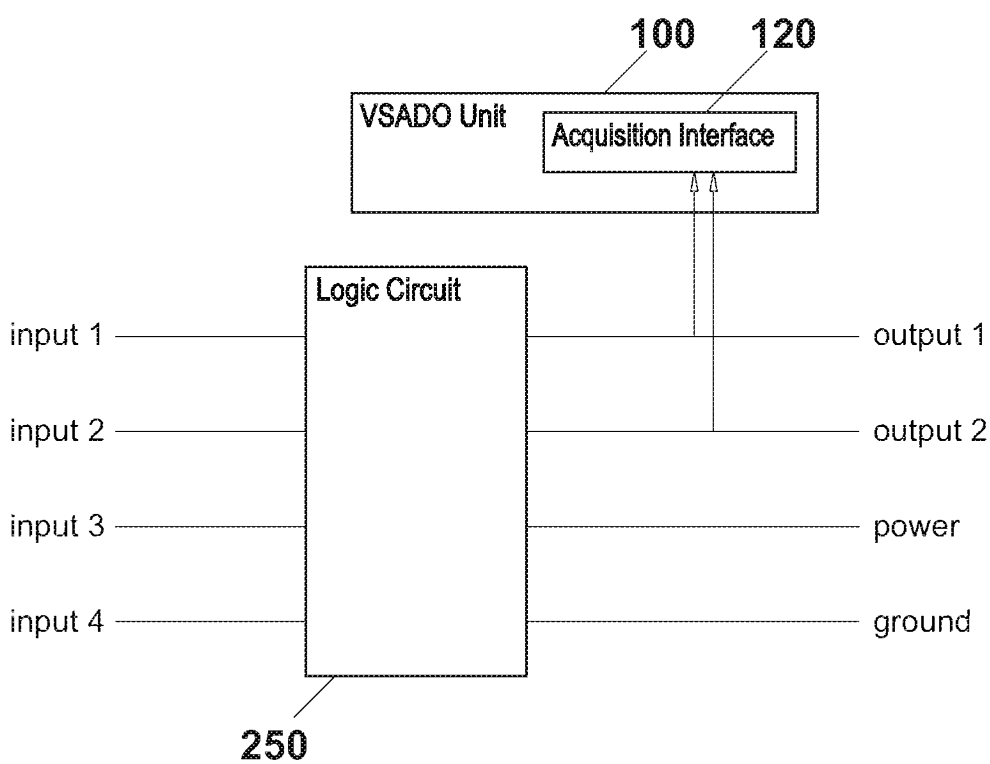

100
120
VSADO Unit
Acquisition Interface
Logic Circuit
input 1
input 2
input 3
input 4
output 1
output 2
power
ground
250

FIG. 4B

526

Instruction Set

Function (Parameter1, Parameter2, Parameter3, ...)

FIG. 5A

526

Instruction Set

SELECT Col1, Col2 ... FROM Tbl1, Tbl2 ... ;

FIG. 5B

526

Instruction Set

iconst_Value

FIG. 5C

526

Instruction Set

add Register1 Register2

FIG. 5D

526

Instruction Set

000000 00001 00010 00110 00000 100000

FIG. 5E

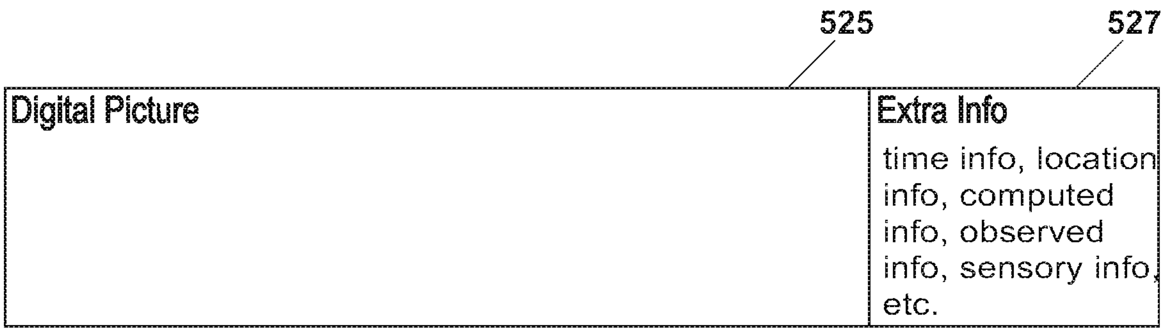

525
527
Digital Picture
Extra Info
time info, location info, computed info, observed info, sensory info, etc.

FIG. 6A

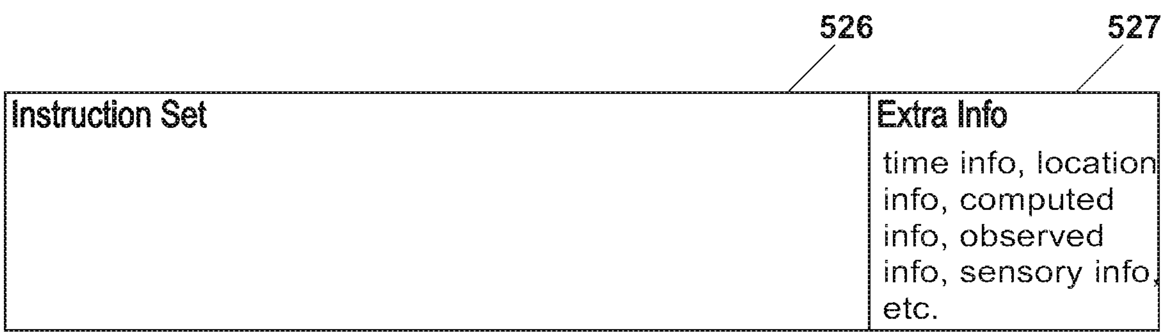

526
527
Instruction Set
Extra Info
time info, location info, computed info, observed info, sensory info, etc.

FIG. 6B

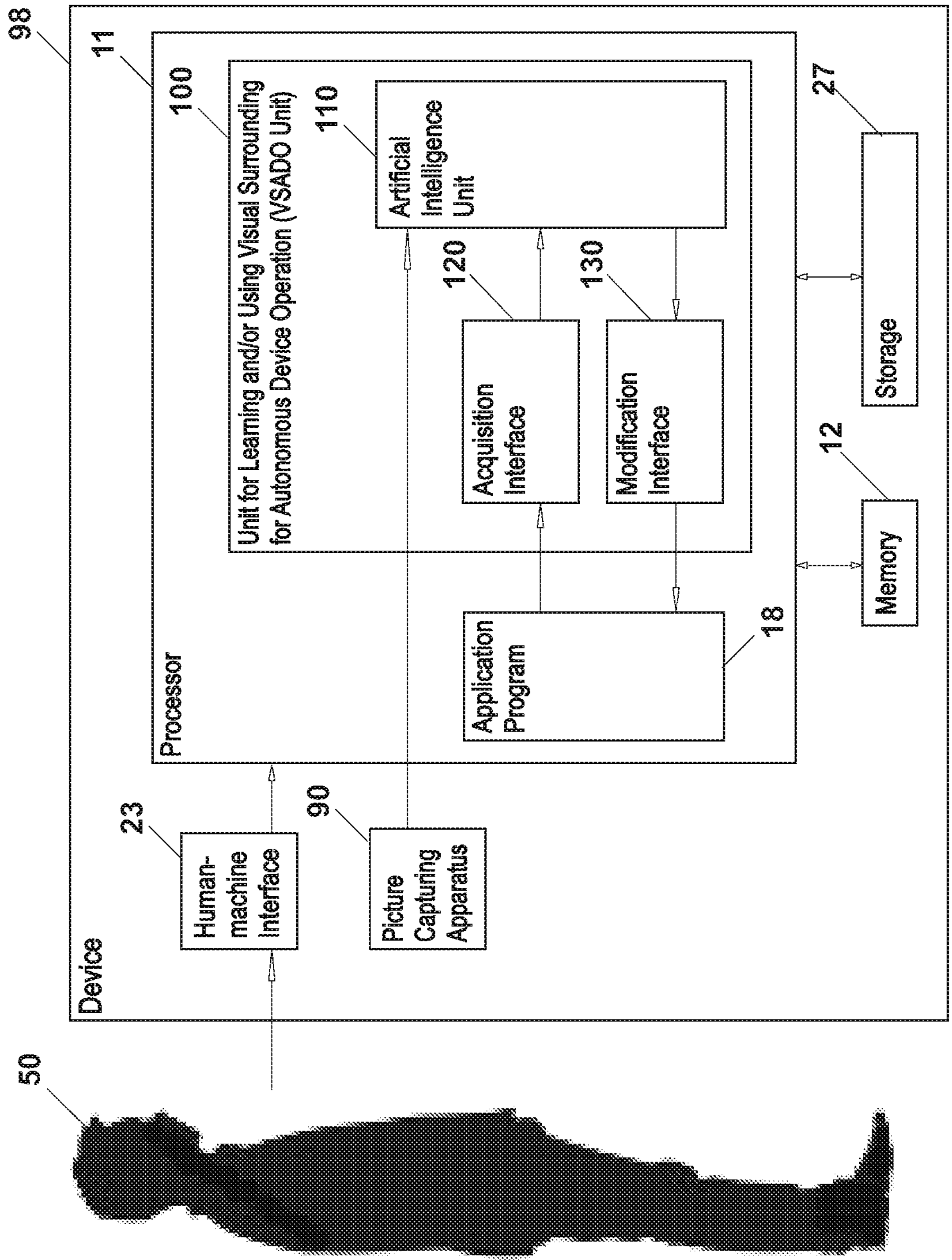

98
Device
11
Processor
100
Unit for Learning and/or Using Visual Surrounding for Autonomous Device Operation (VSADO Unit)
110
Artificial Intelligence Unit
120
Acquisition Interface
130
Modification Interface
Application Program
18
27
Storage
12
Memory
23
Human-machine Interface
90
Picture Capturing Apparatus
50

FIG. 7

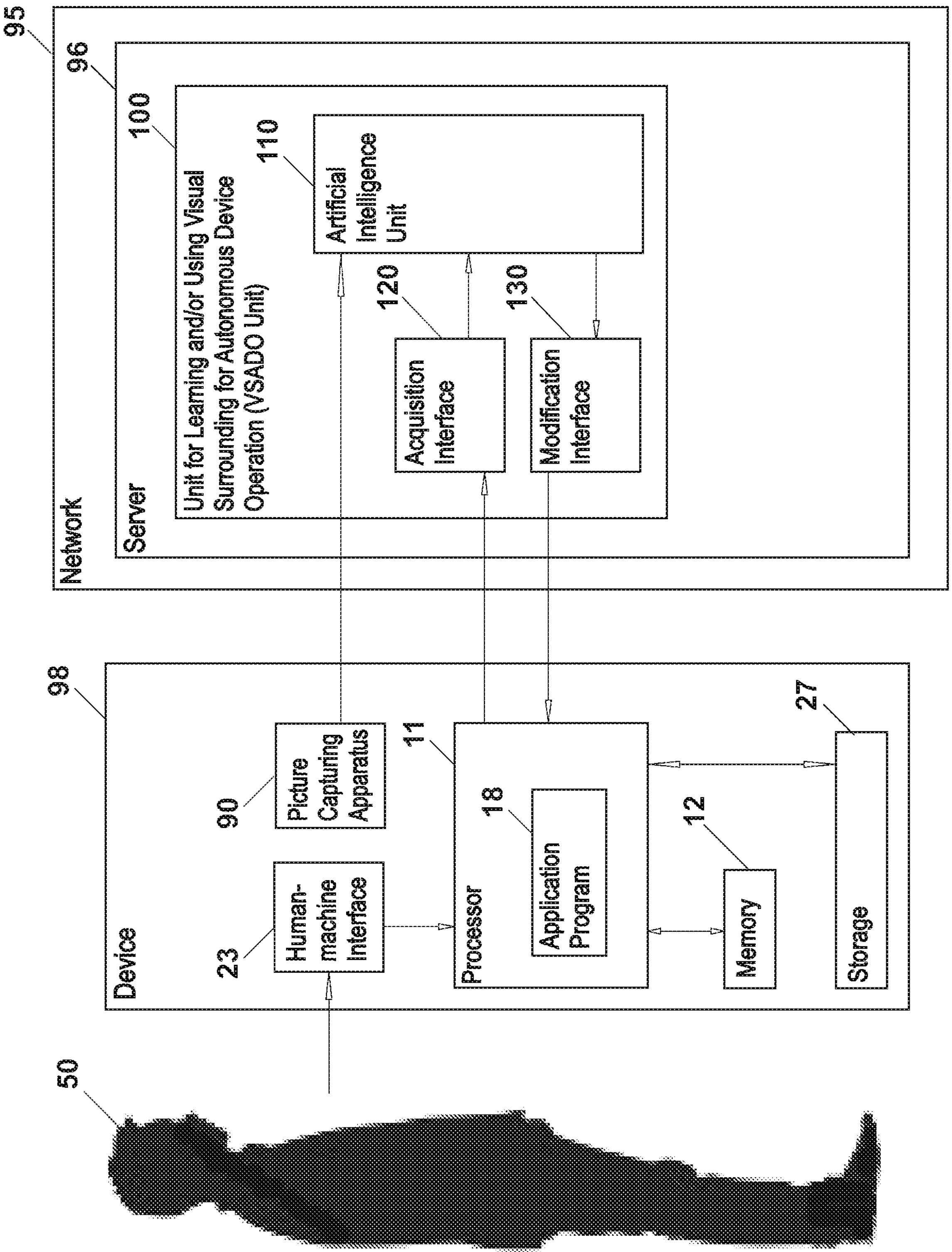

95
96
100
110
Network
Server
Unit for Learning and/or Using Visual Surrounding for Autonomous Device Operation (VSADO Unit)
Artificial Intelligence Unit
120
130
Acquisition Interface
Modification Interface
98
Device
90
Picture Capturing Apparatus
23
Human-machine Interface
11
Processor
18
Application Program
12
Memory
27
Storage
50

FIG. 8

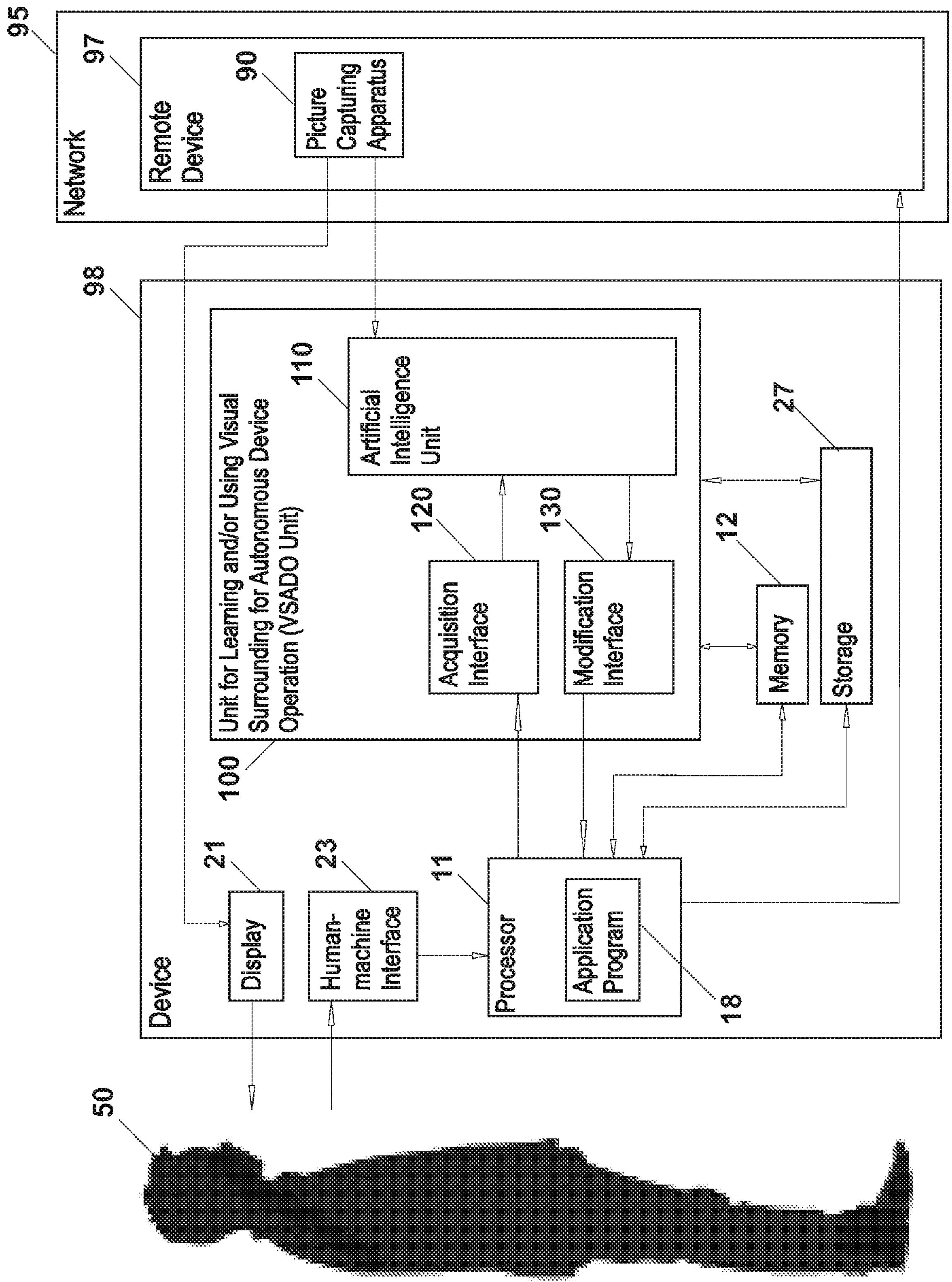

95
Network
97
Remote Device
90
Picture Capturing Apparatus
98
Device
100
Unit for Learning and/or Using Visual Surrounding for Autonomous Device Operation (VSADO Unit)
110
Artificial Intelligence Unit
120
Acquisition Interface
130
Modification Interface
12
Memory
27
Storage
21
Display
23
Human-machine Interface
11
Processor
Application Program
18
50

FIG. 9

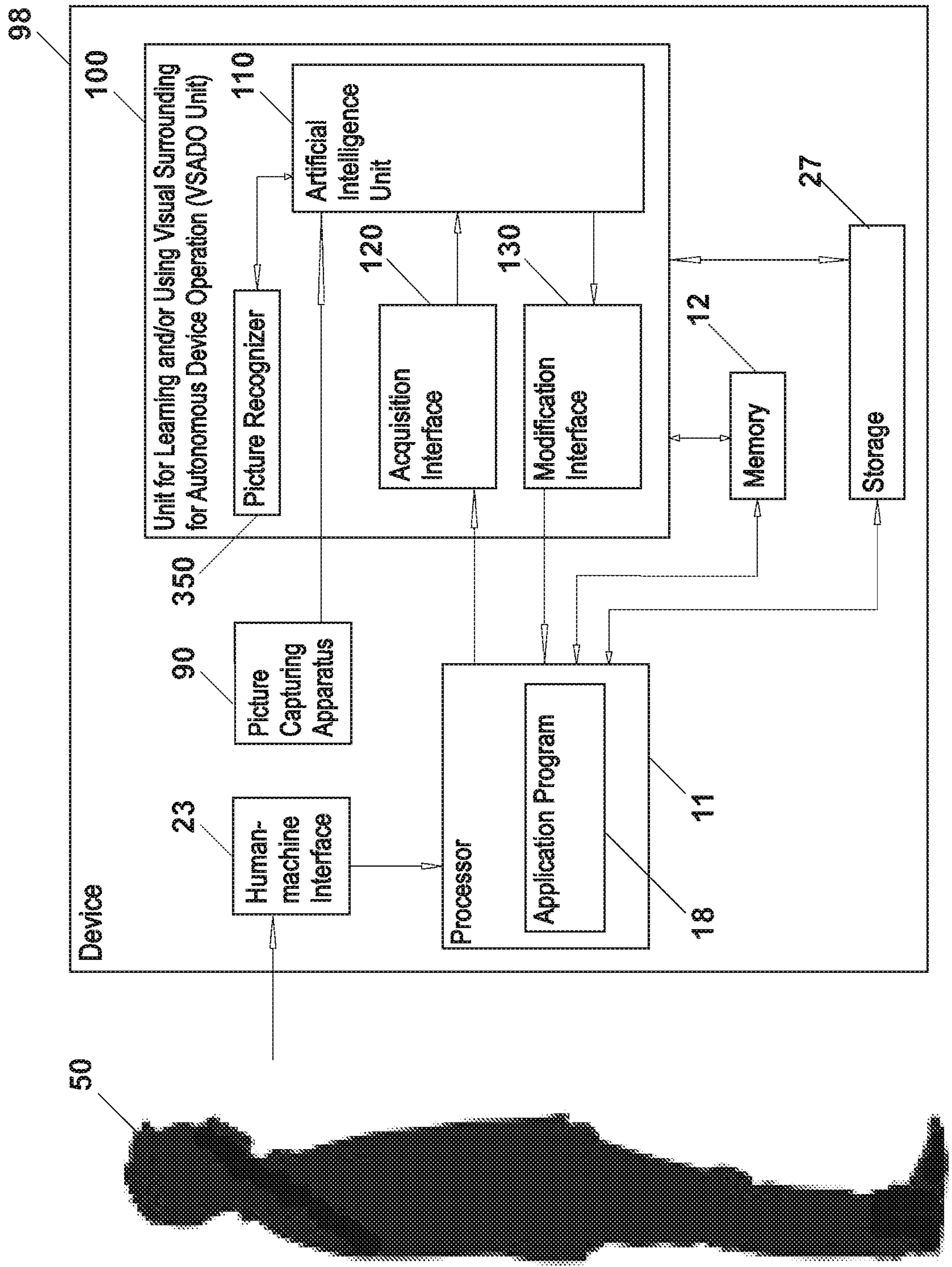

98
Device
100
Unit for Learning and/or Using Visual Surrounding for Autonomous Device Operation (VSADO Unit)
110
Artificial Intelligence Unit
Picture Recognizer
350
120
Acquisition Interface
130
Modification Interface
12
Memory
27
Storage
90
Picture Capturing Apparatus
23
Human-machine Interface
Processor
Application Program
11
18
50

FIG. 10

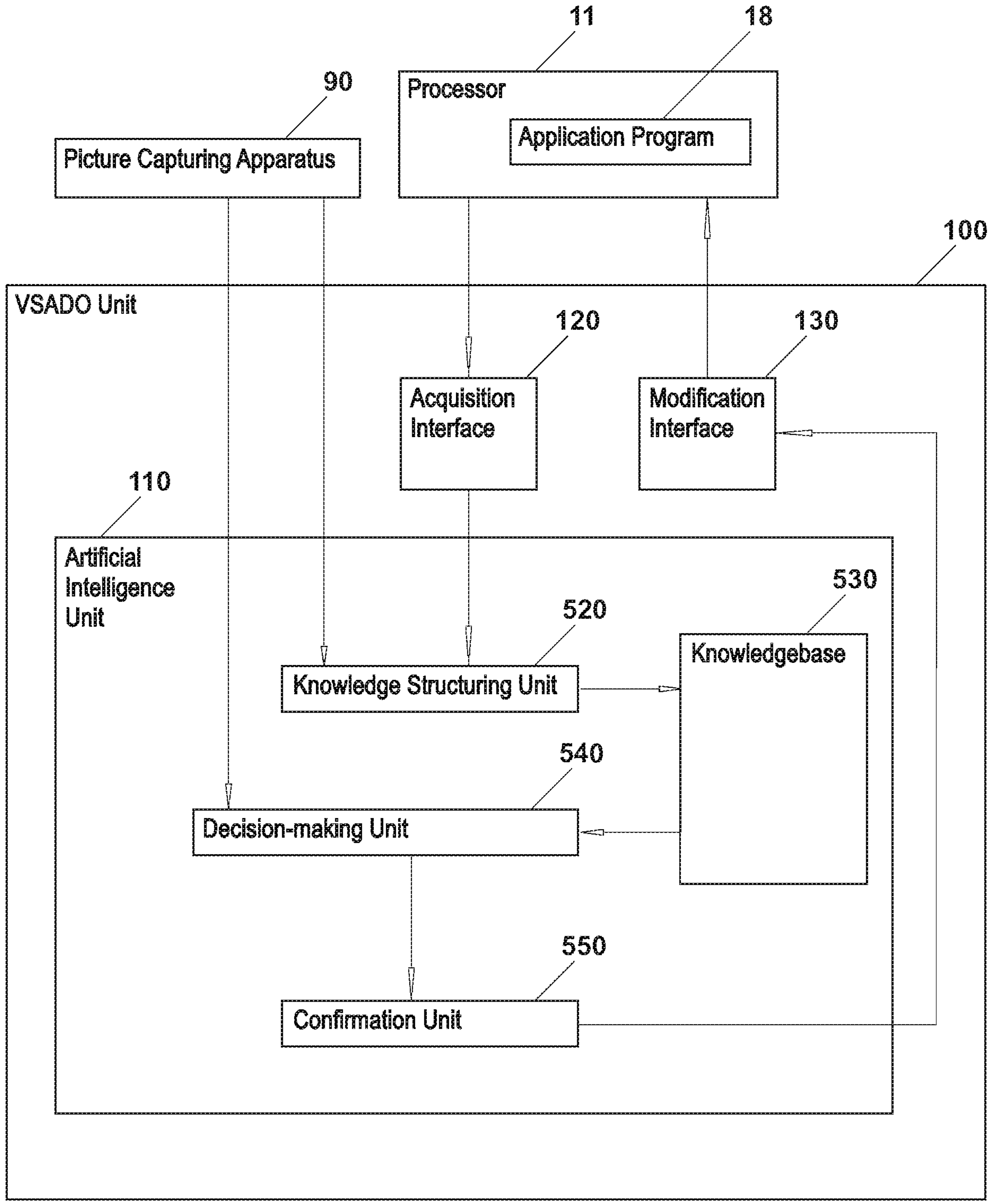
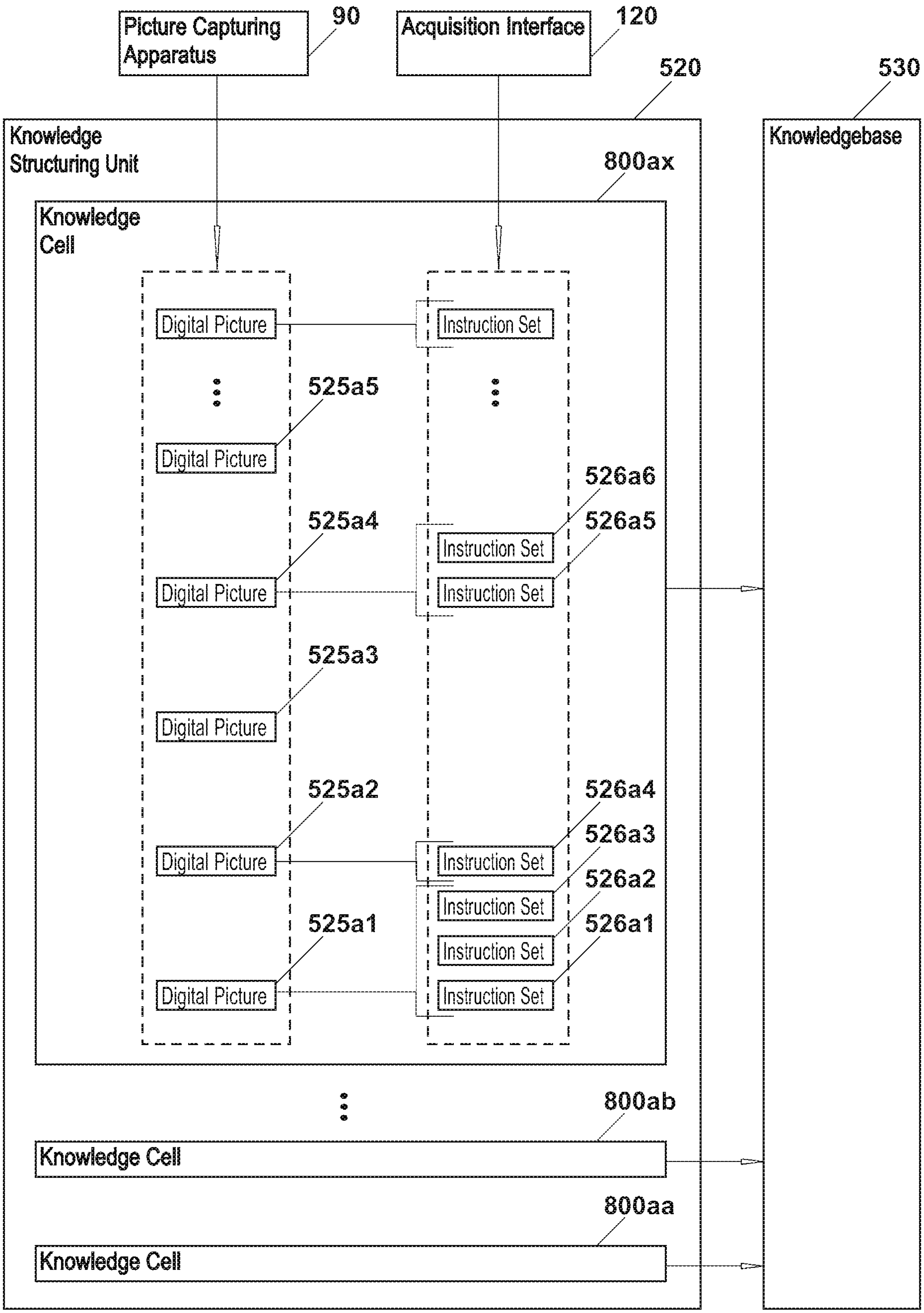
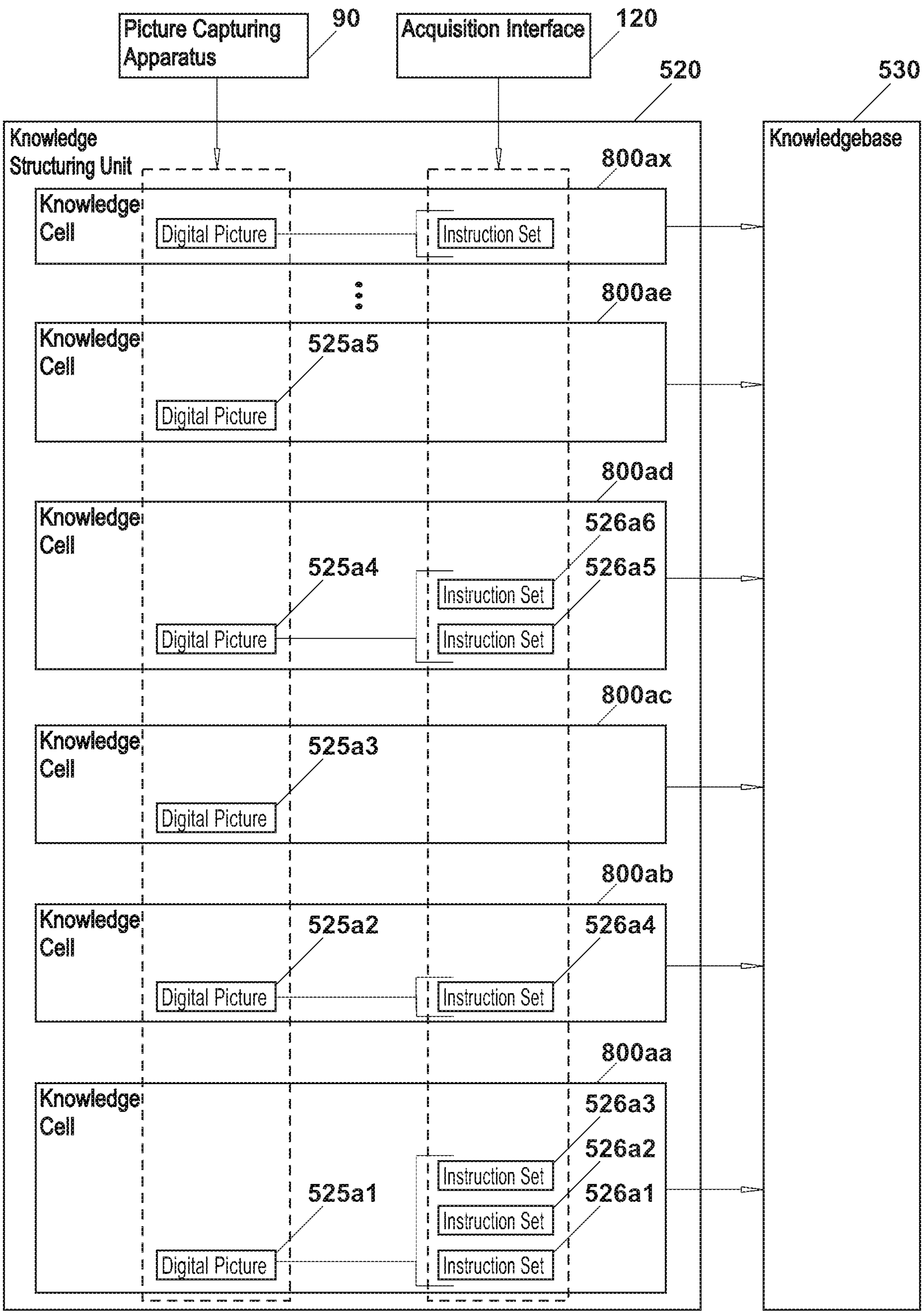

11
18
90
Processor
Application Program
Picture Capturing Apparatus
100
VSADO Unit
120
130
Acquisition Interface
Modification Interface
110
Artificial Intelligence Unit
530
520
Knowledgebase
Knowledge Structuring Unit
540
Decision-making Unit
550
Confirmation Unit

FIG. 11



Picture Capturing Apparatus
90
Acquisition Interface
120
520
530
Knowledge Structuring Unit
800ax
Knowledgebase
Knowledge Cell
Digital Picture
Instruction Set
525a5
Digital Picture
526a6
525a4
526a5
Instruction Set
Digital Picture
Instruction Set
525a3
Digital Picture
525a2
526a4
526a3
Digital Picture
Instruction Set
526a2
Instruction Set
525a1
526a1
Instruction Set
Digital Picture
Instruction Set
800ab
Knowledge Cell
800aa
Knowledge Cell

FIG. 12



Picture Capturing Apparatus
90
Acquisition Interface
120
520
530
Knowledge Structuring Unit
Knowledgebase
800ax
Knowledge Cell
Digital Picture
Instruction Set
800ae
Knowledge Cell
525a5
Digital Picture
800ad
Knowledge Cell
526a6
525a4
526a5
Instruction Set
Digital Picture
Instruction Set
800ac
Knowledge Cell
525a3
Digital Picture
800ab
Knowledge Cell
525a2
526a4
Digital Picture
Instruction Set
800aa
Knowledge Cell
526a3
526a2
Instruction Set
525a1
526a1
Instruction Set
Digital Picture
Instruction Set

FIG. 13

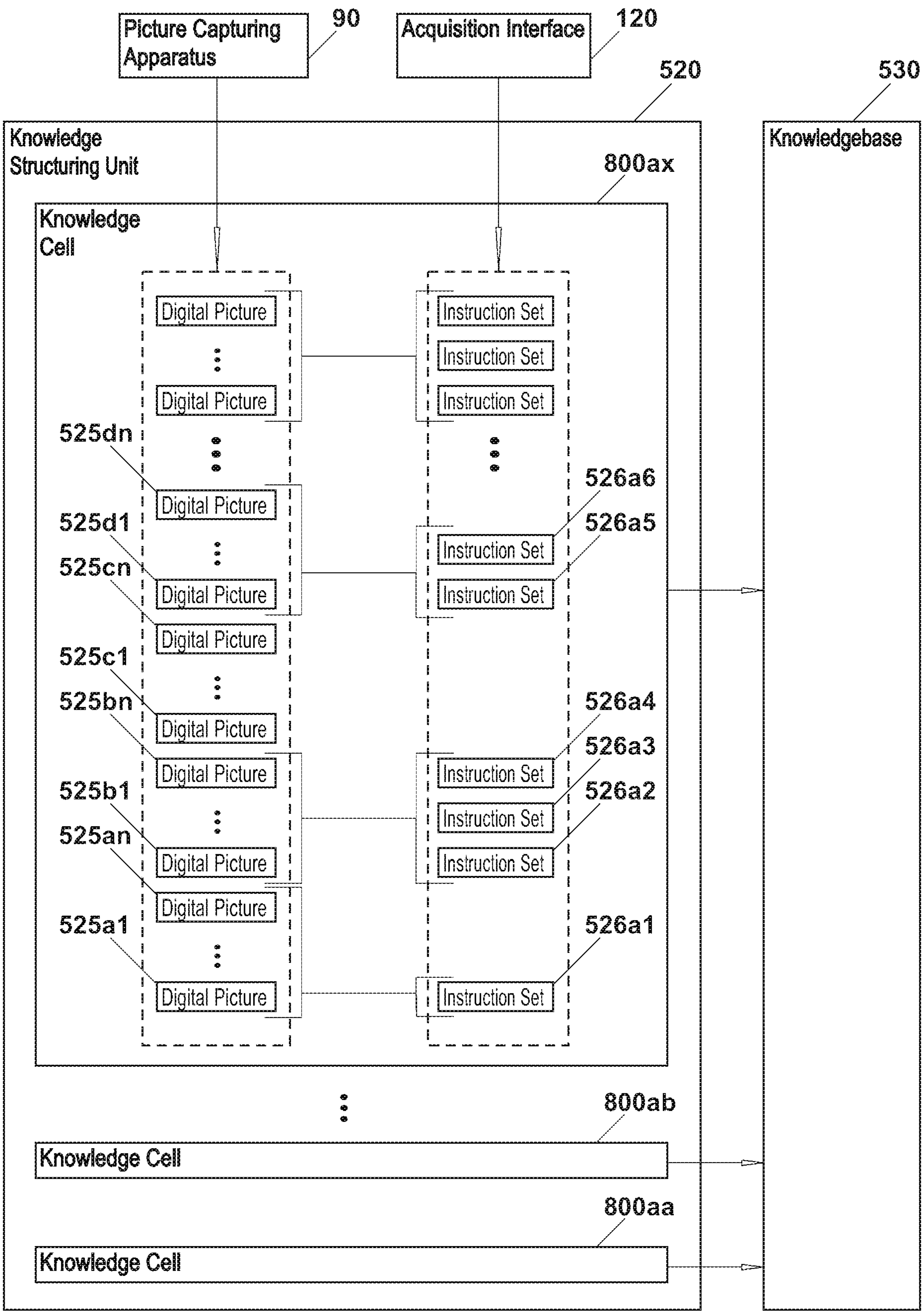

Picture Capturing Apparatus
90
Acquisition Interface
120
520
530
Knowledge Structuring Unit
800ax
Knowledgebase
Knowledge Cell
Digital Picture
Instruction Set
525dn
525d1
525cn
525c1
525bn
525b1
525an
525a1
526a6
526a5
526a4
526a3
526a2
526a1
800ab
Knowledge Cell
800aa
Knowledge Cell

FIG. 14

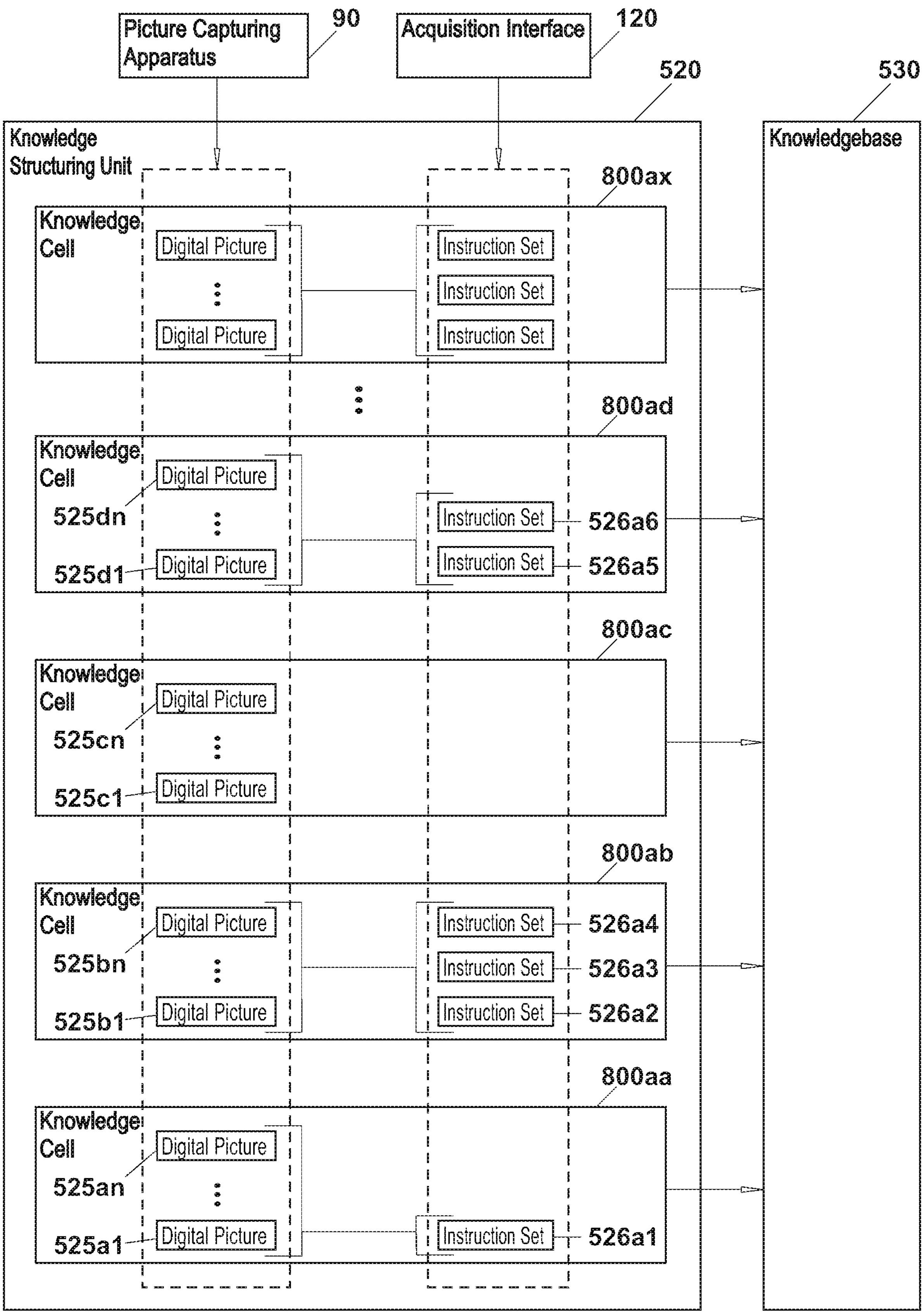

Picture Capturing Apparatus
90
Acquisition Interface
120
520
530
Knowledge Structuring Unit
Knowledgebase
800ax
Knowledge Cell
Digital Picture
Digital Picture
Instruction Set
Instruction Set
Instruction Set
800ad
Knowledge Cell
525dn
525d1
Digital Picture
Digital Picture
Instruction Set
526a6
Instruction Set
526a5
800ac
Knowledge Cell
525cn
525c1
Digital Picture
Digital Picture
800ab
Knowledge Cell
525bn
525b1
Digital Picture
Digital Picture
Instruction Set
526a4
Instruction Set
526a3
Instruction Set
526a2
800aa
Knowledge Cell
525an
525a1
Digital Picture
Digital Picture
Instruction Set
526a1

FIG. 15

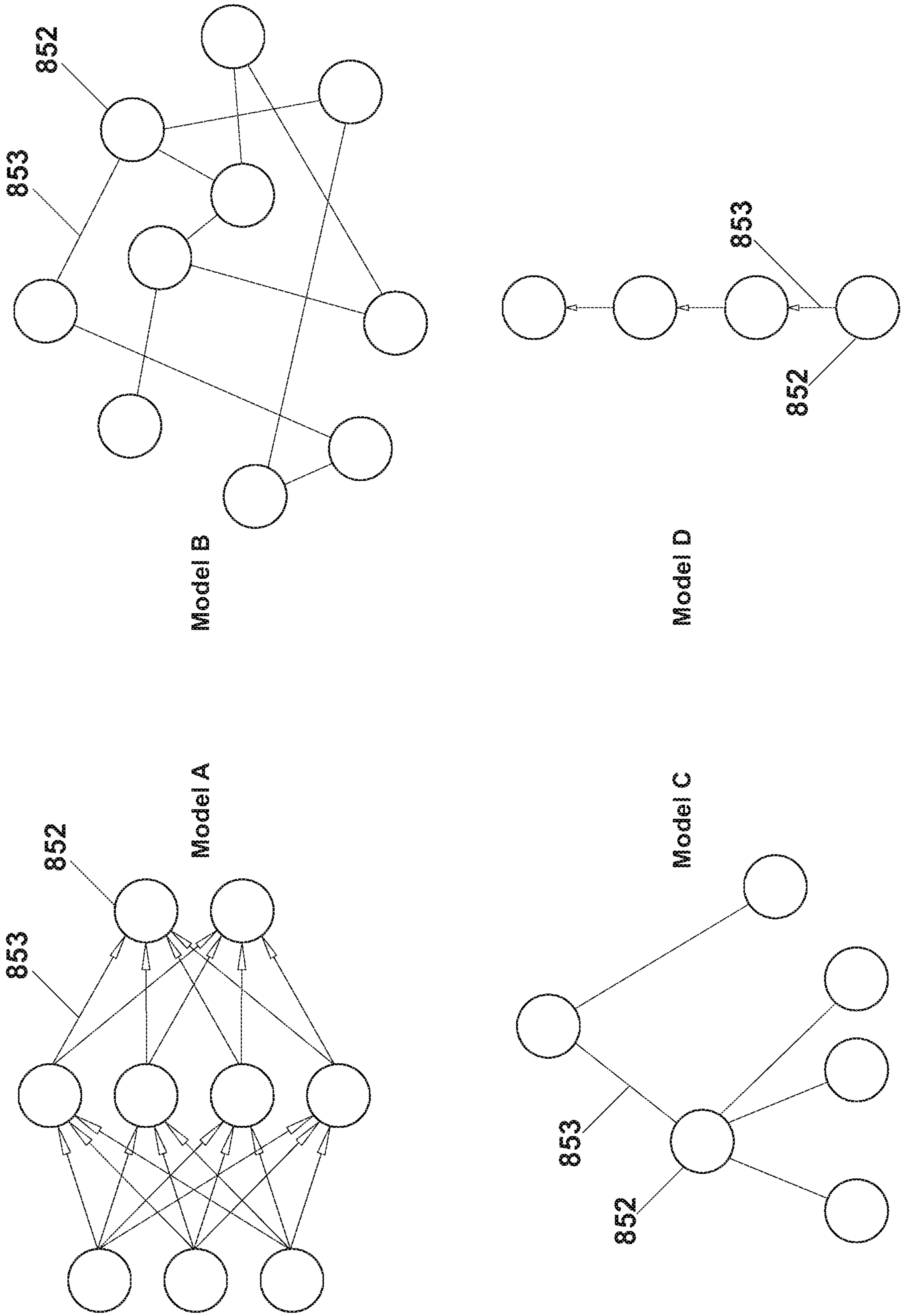


FIG. 16

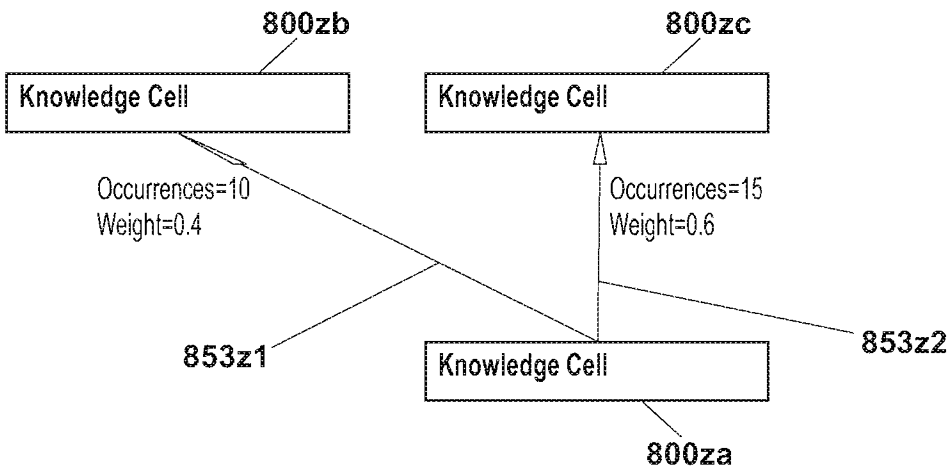


FIG. 17A

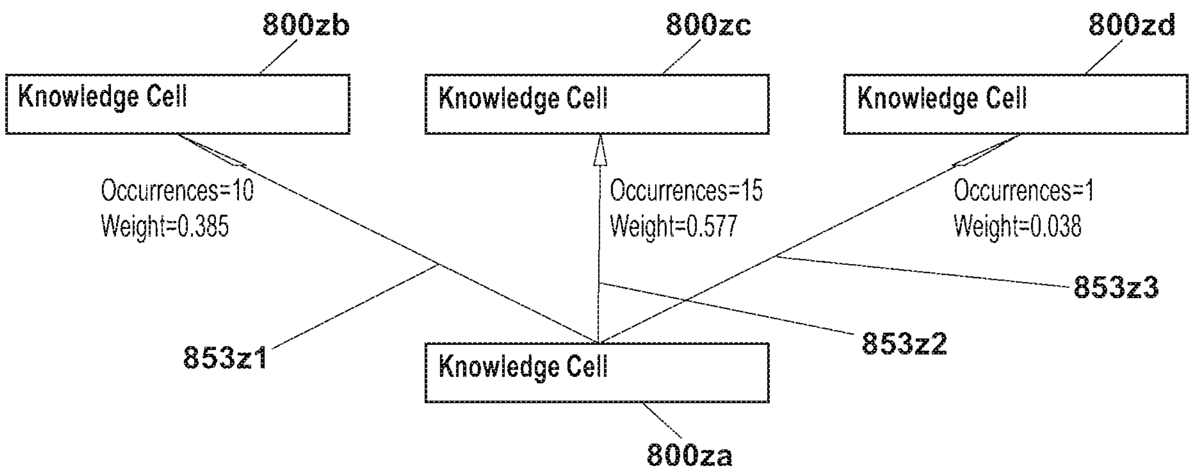


FIG. 17B

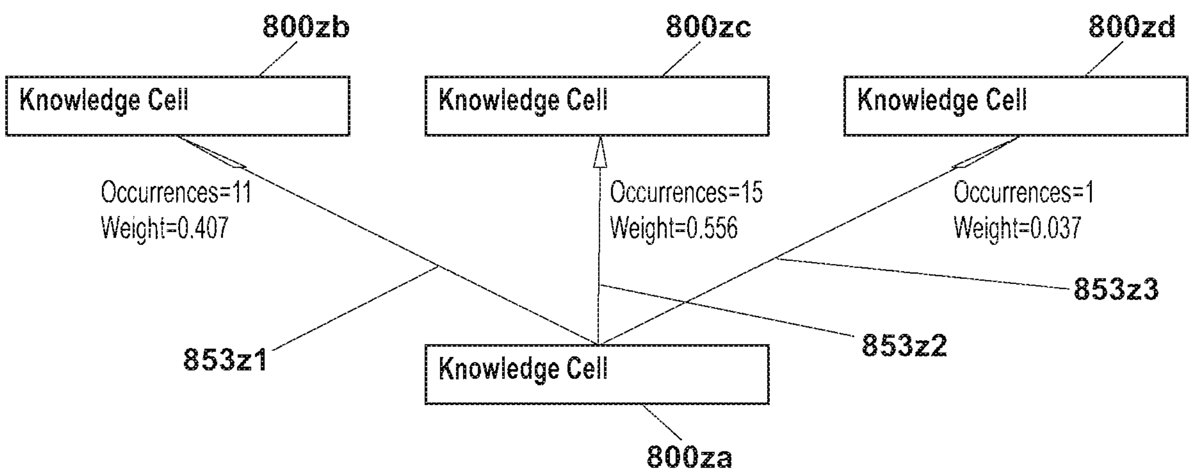


FIG. 17C

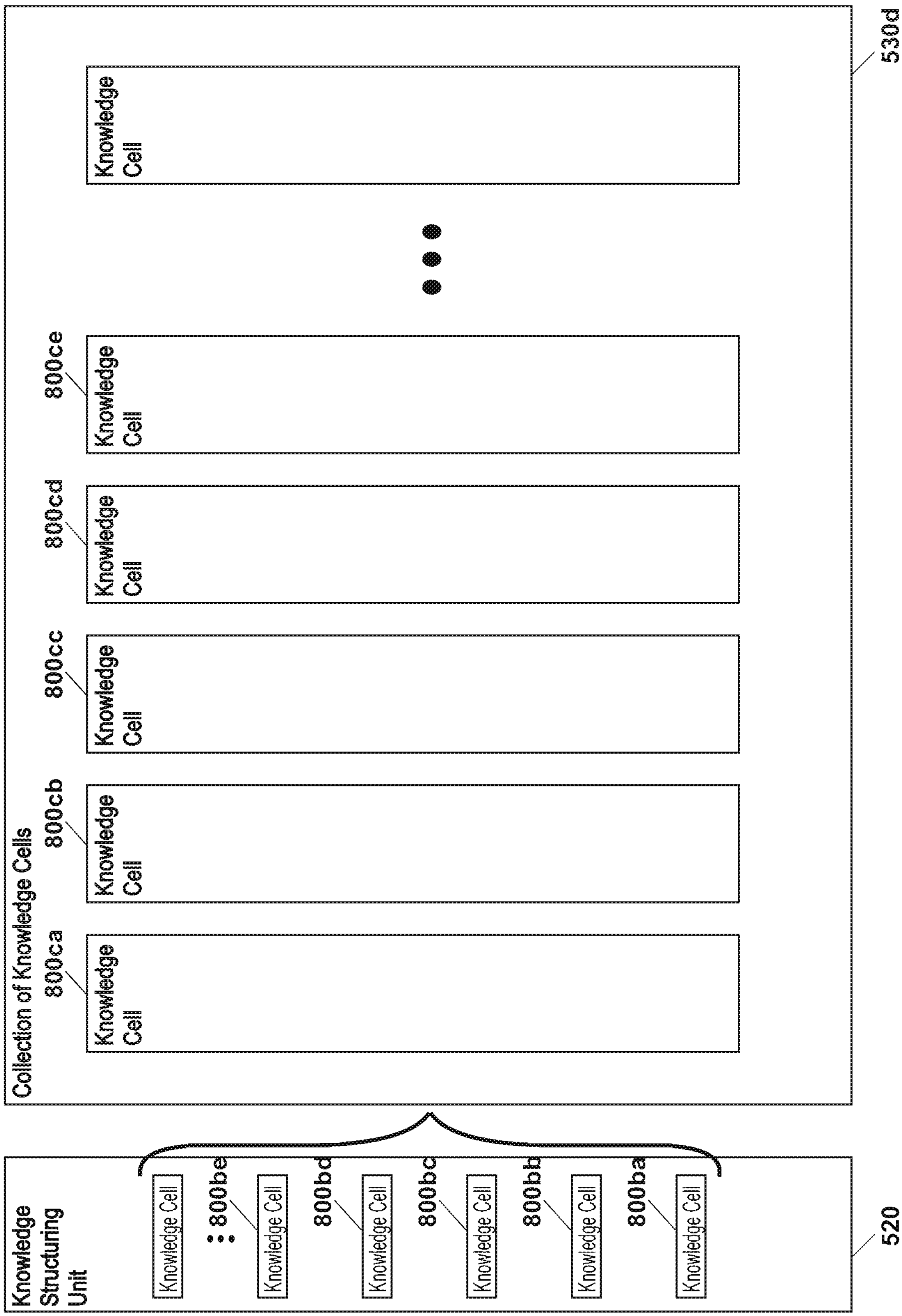

Collection of Knowledge Cells
Knowledge Cell
800ca
800cb
800cc
800cd
800ce
530d
Knowledge Structuring Unit
800ba
800bb
800bc
800bd
800be
520

FIG. 18

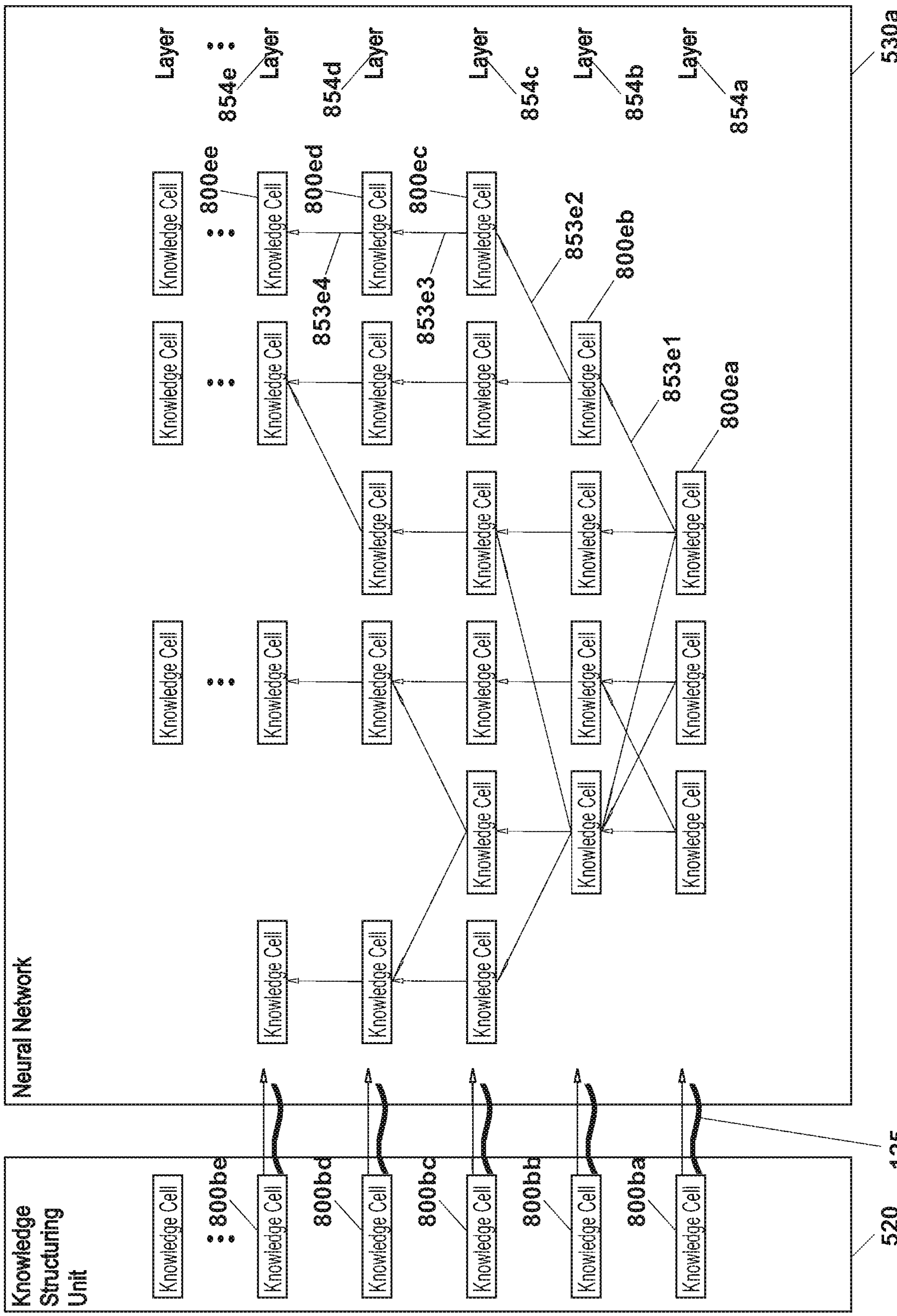

Neural Network
Knowledge Structuring Unit
Knowledge Cell
Layer
854a
854b
854c
854d
854e
800ea
800eb
800ec
800ed
800ee
853e1
853e2
853e3
853e4
800ba
800bb
800bc
800bd
800be
125
520
530a

FIG. 19

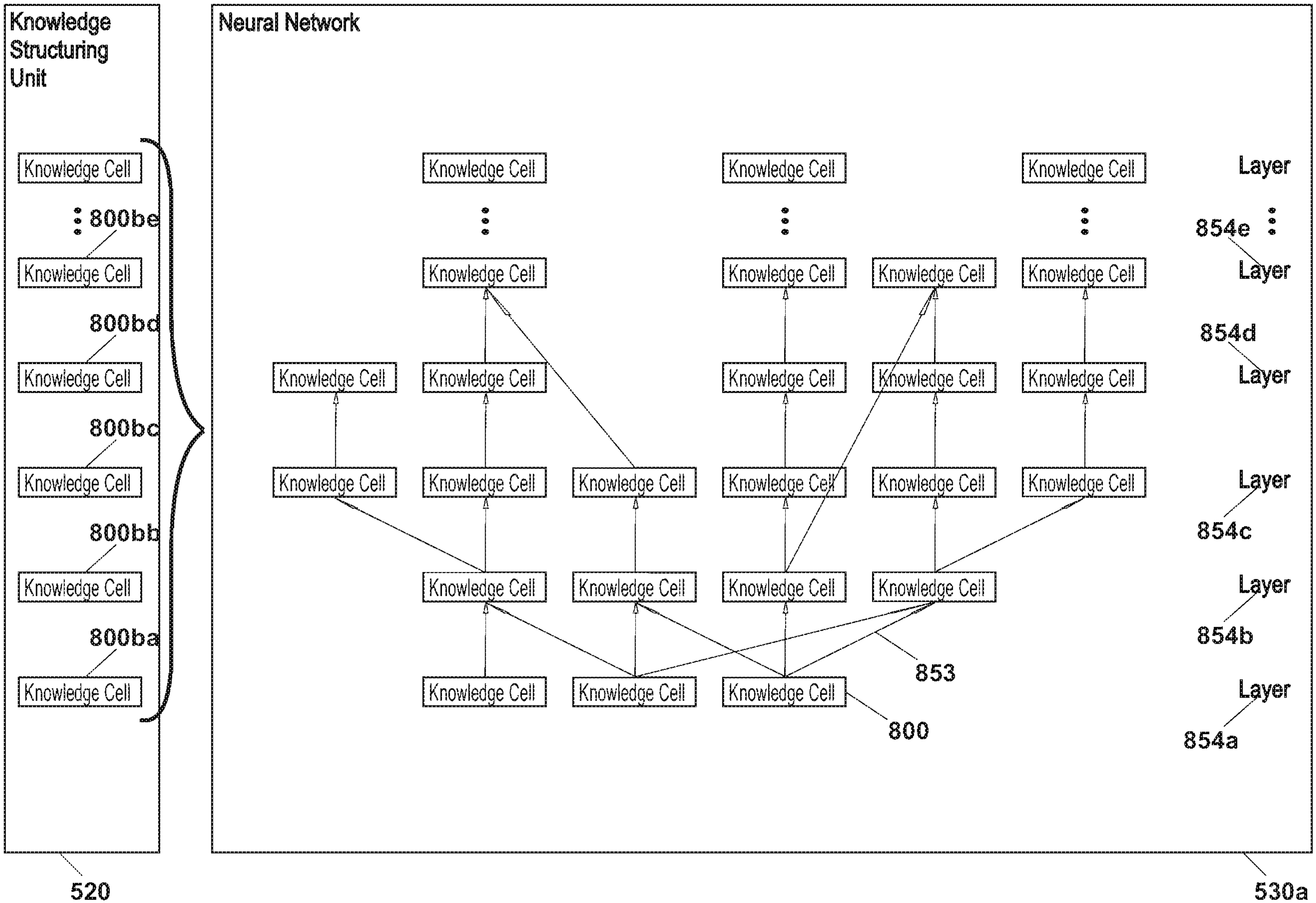

Knowledge Structuring Unit
Knowledge Cell
800be
800bd
800bc
800bb
800ba
520
Neural Network
Layer
854e
854d
854c
854b
854a
853
800
530a

FIG. 20

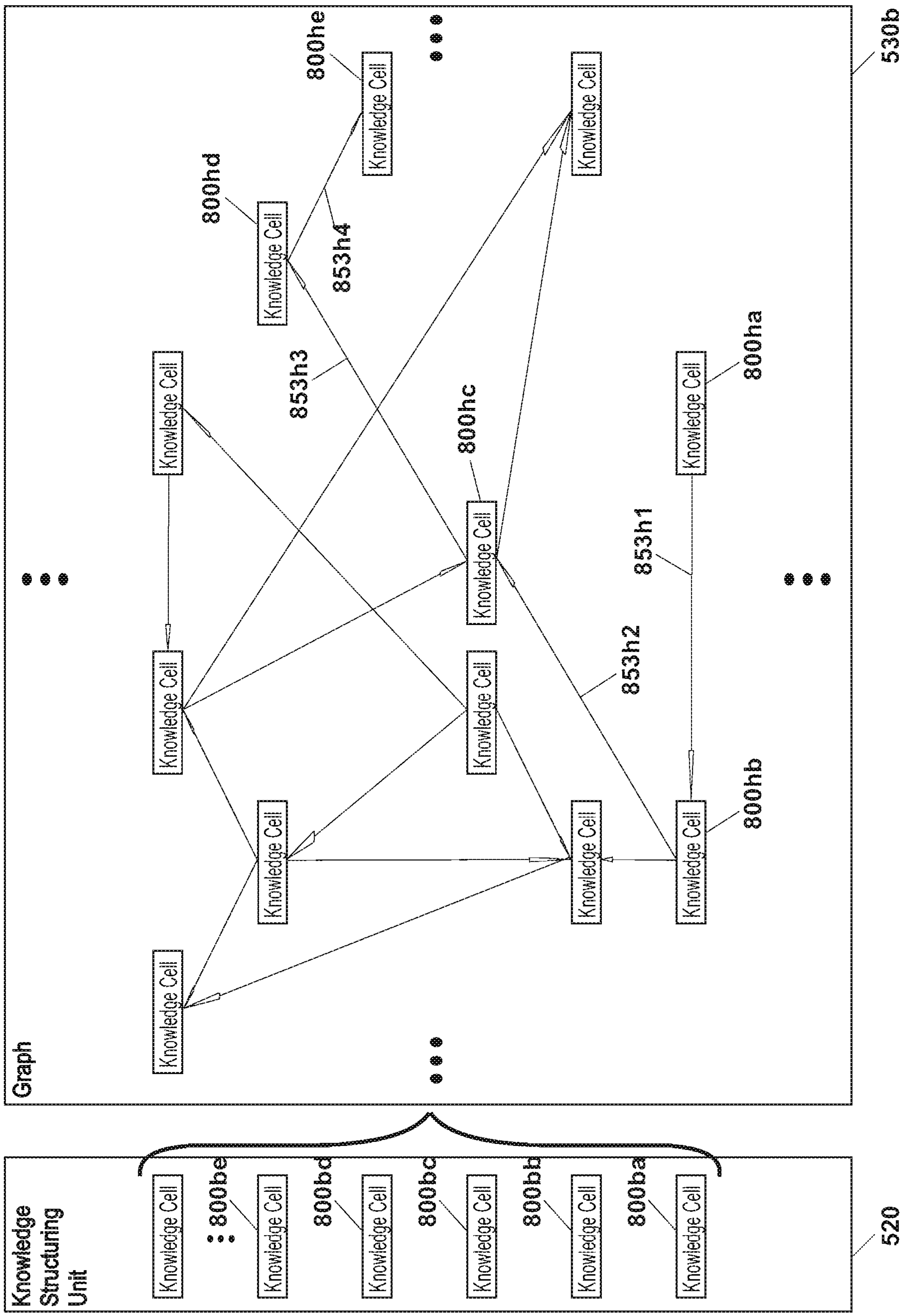
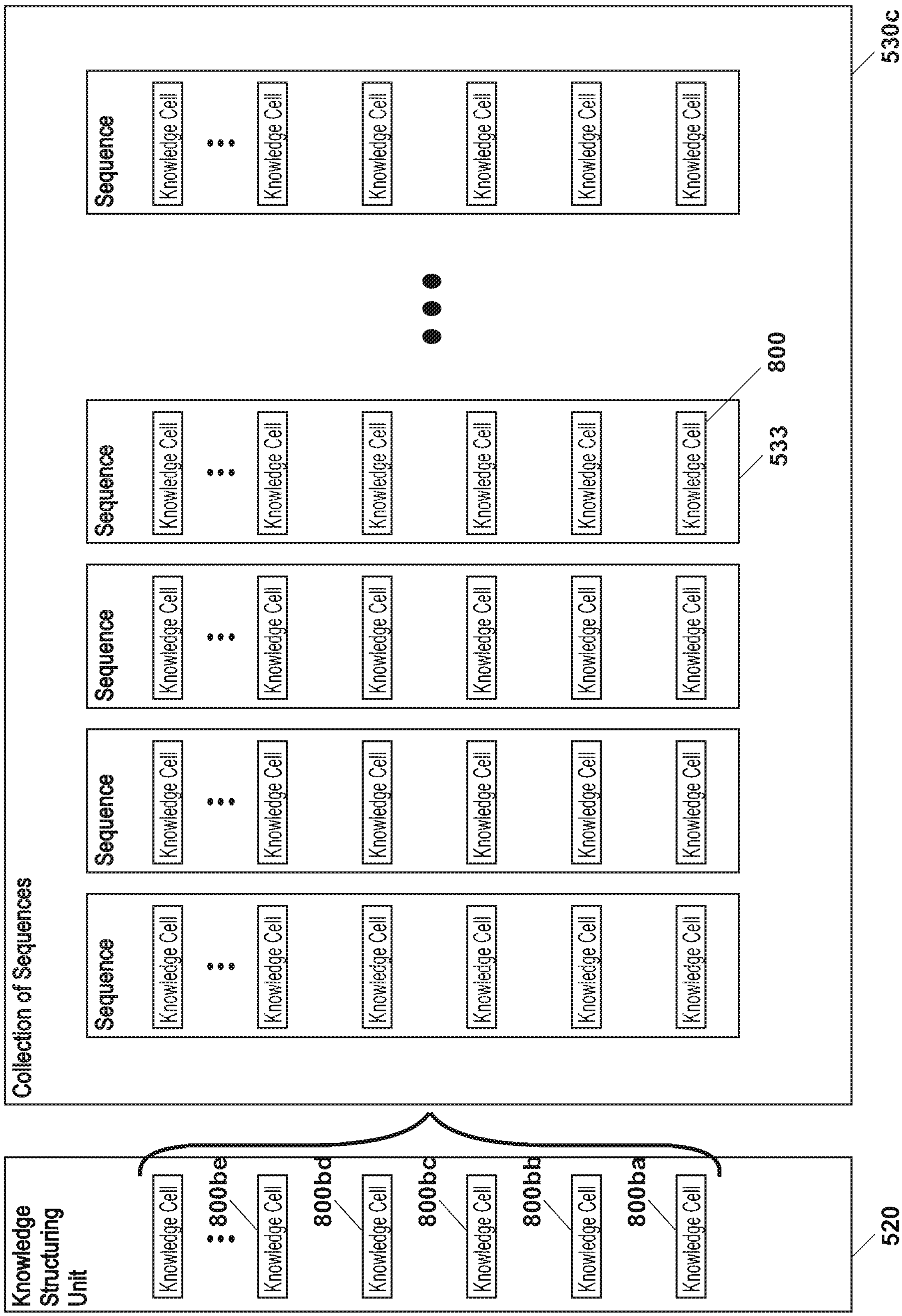
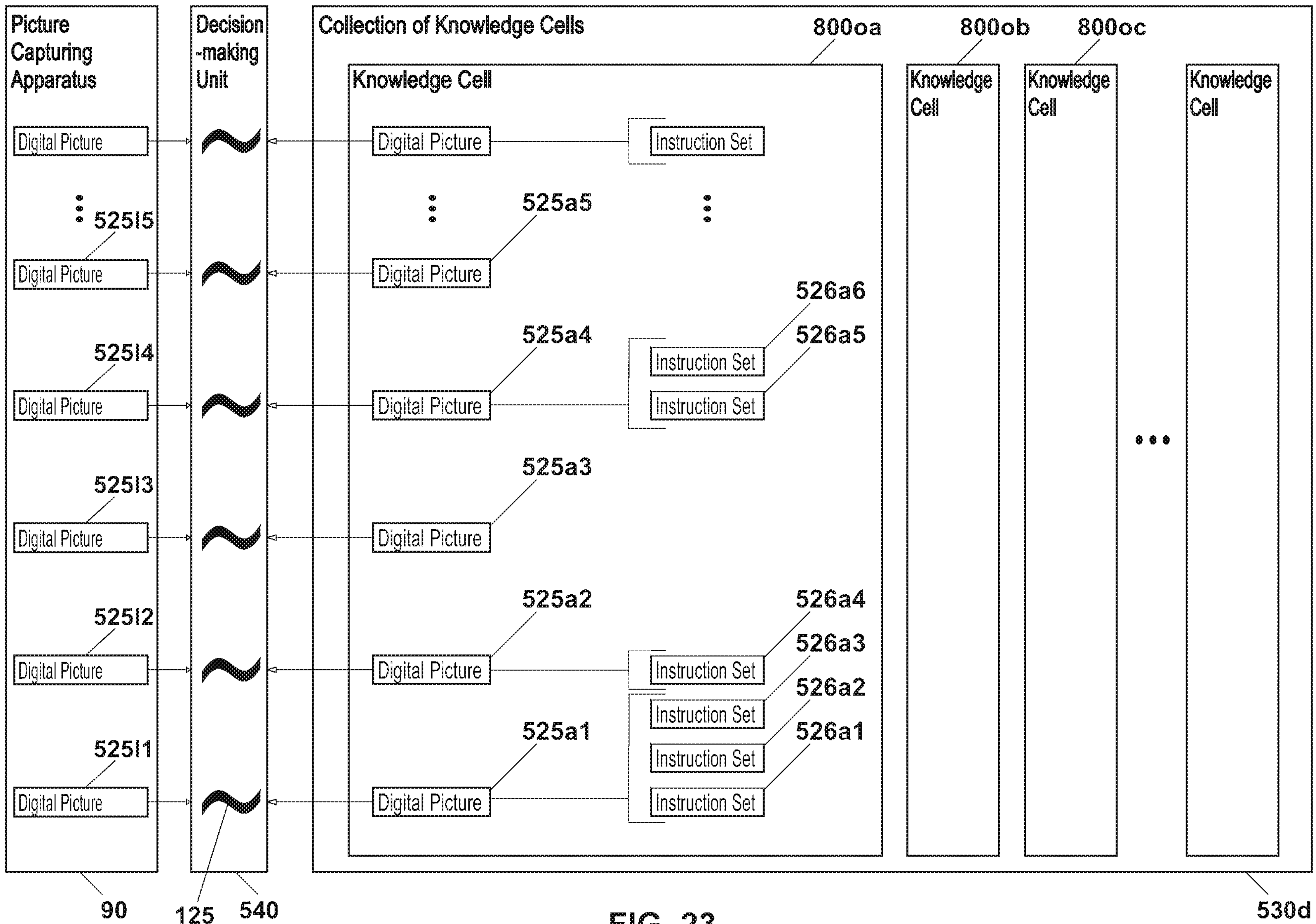

Graph
Knowledge Cell
800ha
800hb
800hc
800hd
800he
853h1
853h2
853h3
853h4
530b
Knowledge Structuring Unit
800ba
800bb
800bc
800bd
800be
520

FIG. 21



Knowledge Structuring Unit
Knowledge Cell
800be
800bd
800bc
800bb
800ba
520
Collection of Sequences
Sequence
Knowledge Cell
533
800
530c

FIG. 22



Picture Capturing Apparatus
Digital Picture
525l5
525l4
525l3
525l2
525l1
90
Decision -making Unit
125
540
Collection of Knowledge Cells
800oa
800ob
800oc
Knowledge Cell
Digital Picture
Instruction Set
525a5
525a4
525a3
525a2
525a1
526a6
526a5
526a4
526a3
526a2
526a1
Knowledge Cell
Knowledge Cell
Knowledge Cell
530d

FIG. 23

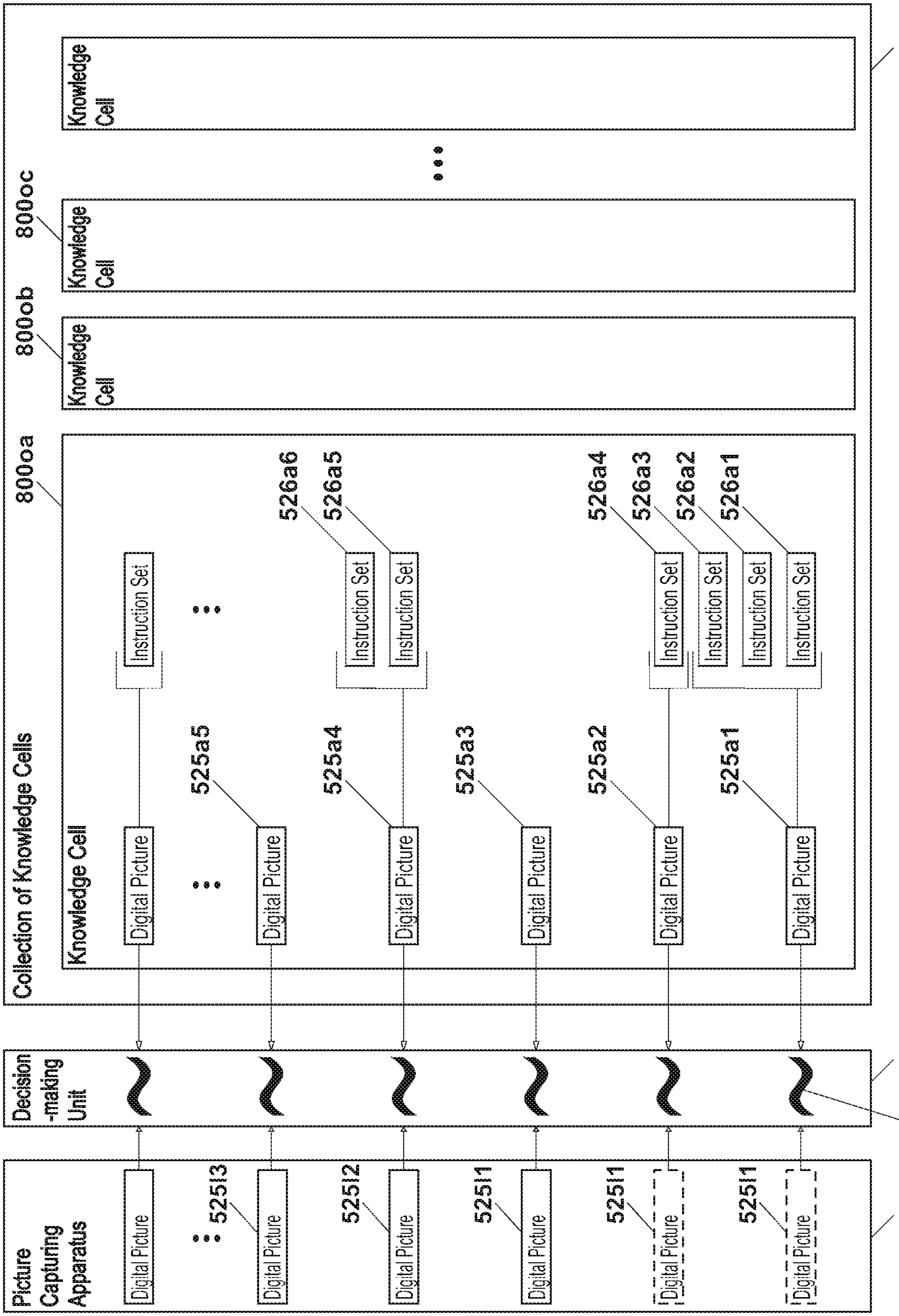

Picture Capturing Apparatus
Digital Picture
525l3
525l2
525l1
525l1
525l1
90
Decision -making Unit
125
540
Collection of Knowledge Cells
Knowledge Cell
525a5
525a4
525a3
525a2
525a1
Instruction Set
526a6
526a5
526a4
526a3
526a2
526a1
800oa
Knowledge Cell
800ob
800oc
530d

FIG. 24

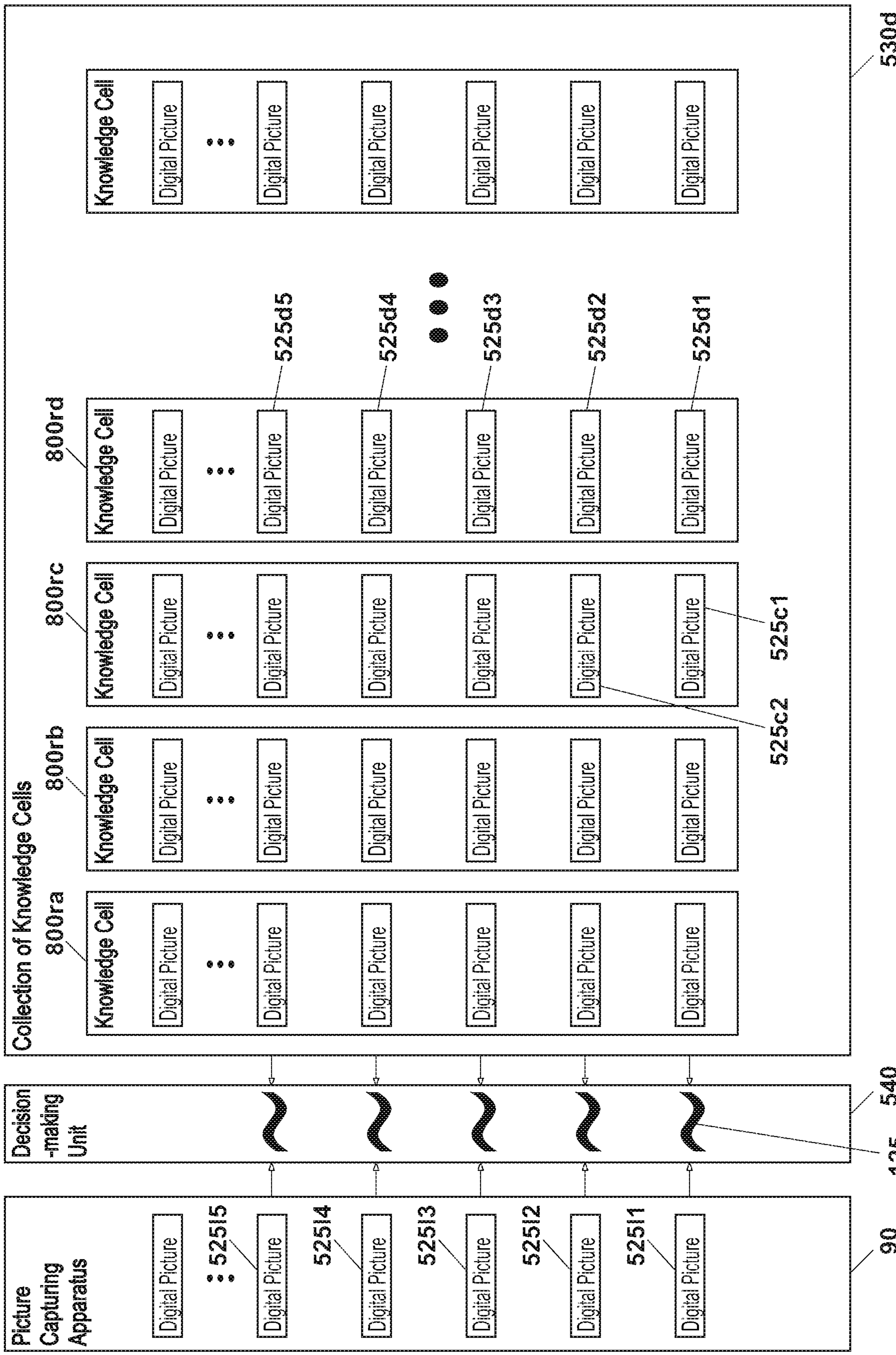

Picture Capturing Apparatus
Digital Picture
525l5
525l4
525l3
525l2
525l1
90
Decision -making Unit
125
540
Collection of Knowledge Cells
800ra
800rb
800rc
800rd
Knowledge Cell
525c1
525c2
525d1
525d2
525d3
525d4
525d5
530d

FIG. 25

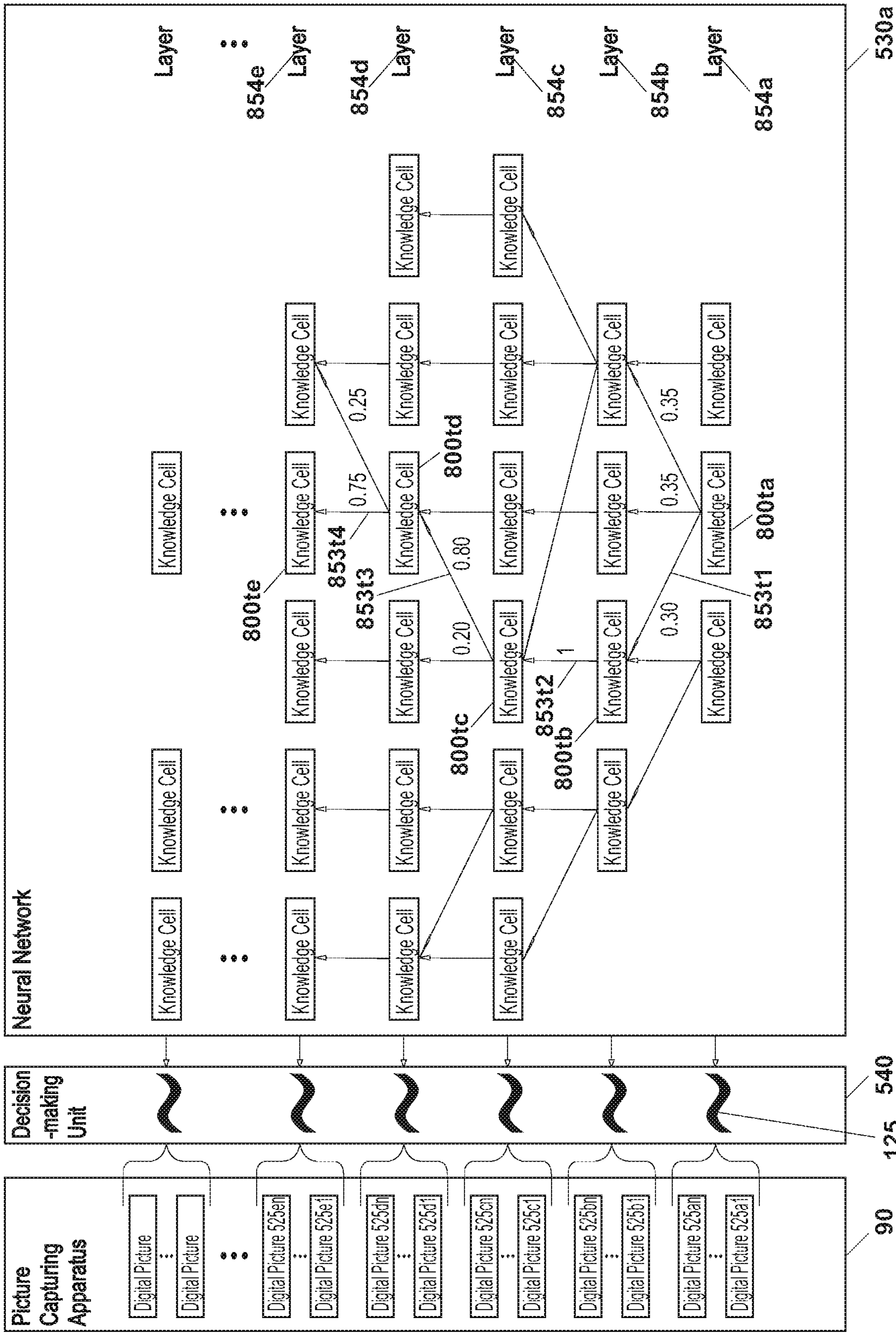

Neural Network
Knowledge Cell
Layer
854a
854b
854c
854d
854e
800ta
800tb
800tc
800td
800te
853t1
853t2
853t3
853t4
0.75
0.25
0.80
0.20
0.35
0.35
0.30
1
530a
Decision
-making
Unit
540
125
Picture
Capturing
Apparatus
90
Digital Picture 525a1
Digital Picture 525an
Digital Picture 525b1
Digital Picture 525bn
Digital Picture 525c1
Digital Picture 525cn
Digital Picture 525d1
Digital Picture 525dn
Digital Picture 525e1
Digital Picture 525en
Digital Picture

FIG. 26

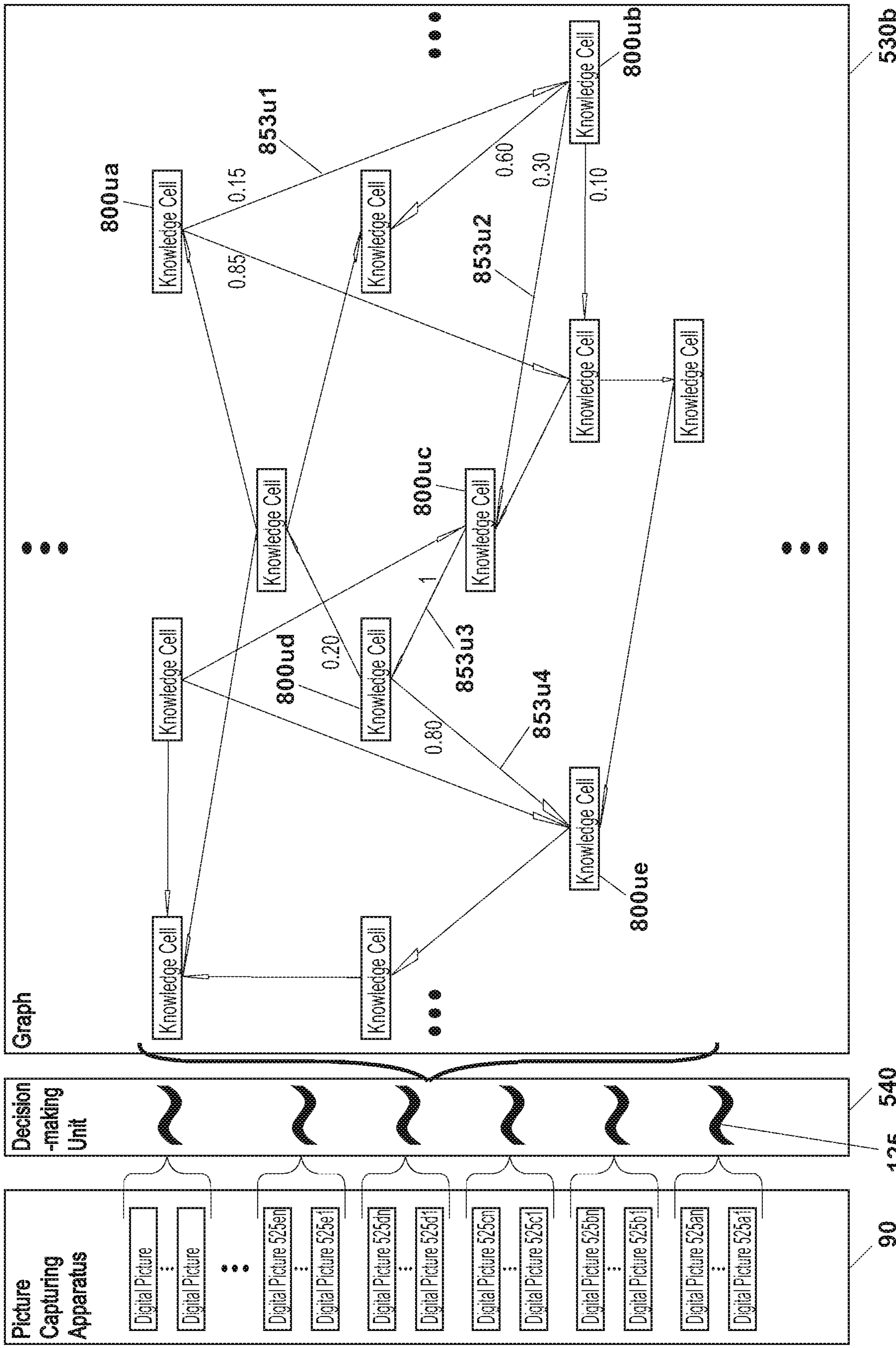

Picture Capturing Apparatus
Digital Picture
Digital Picture
Digital Picture 525en
Digital Picture 525e1
Digital Picture 525dn
Digital Picture 525d1
Digital Picture 525cn
Digital Picture 525c1
Digital Picture 525bn
Digital Picture 525b1
Digital Picture 525an
Digital Picture 525a1
90
Decision-making Unit
540
125
Graph
530b
Knowledge Cell
800ua
800ub
800uc
800ud
800ue
853u1
853u2
853u3
853u4
0.15
0.85
0.60
0.30
0.10
0.20
0.80
1

FIG. 27

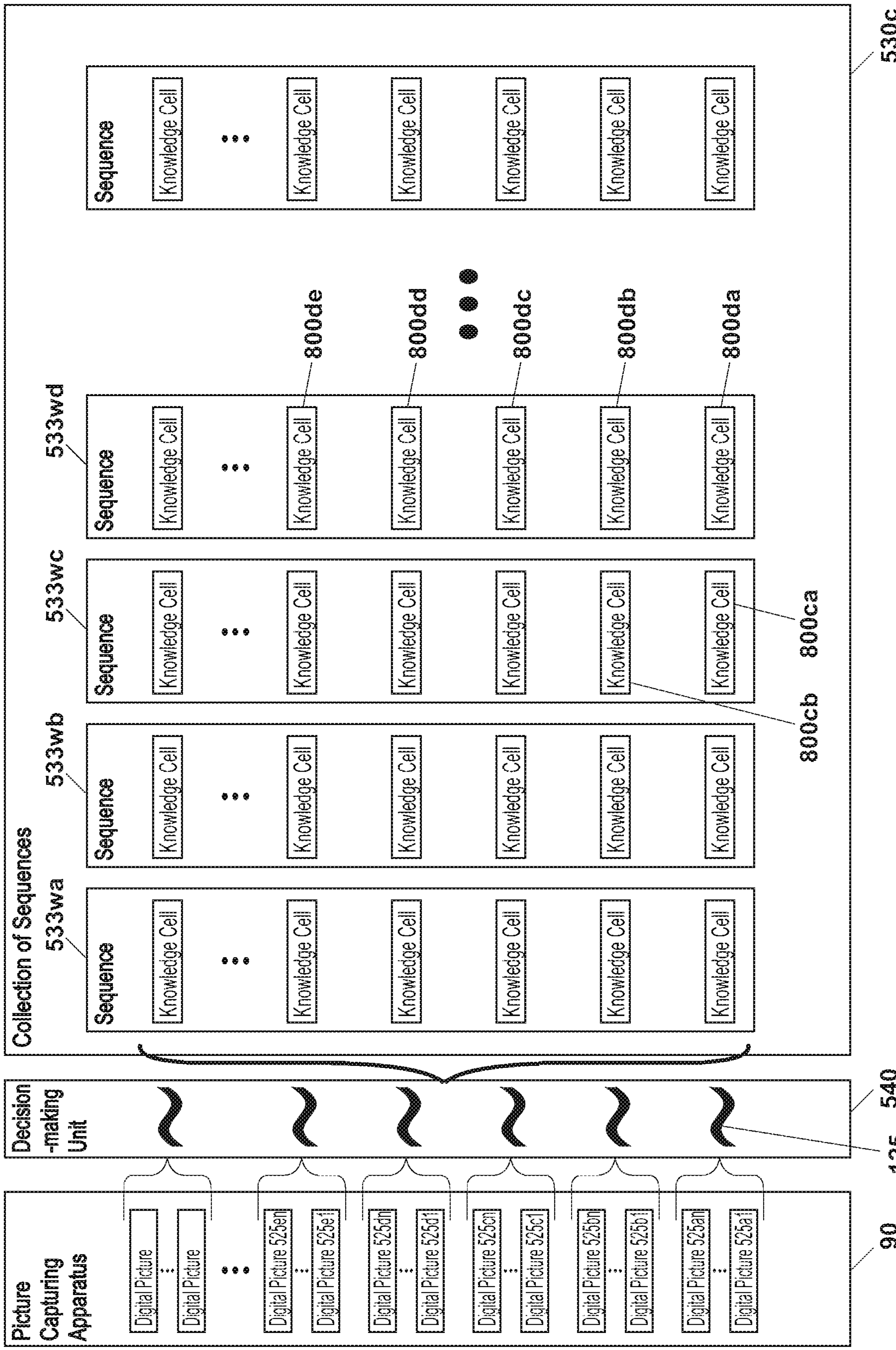

Picture Capturing Apparatus
90
Digital Picture
Digital Picture
Digital Picture 525en
Digital Picture 525e1
Digital Picture 525dn
Digital Picture 525d1
Digital Picture 525cn
Digital Picture 525c1
Digital Picture 525bn
Digital Picture 525b1
Digital Picture 525an
Digital Picture 525a1
Decision -making Unit
540
125
Collection of Sequences
530c
533wa
533wb
533wc
533wd
Sequence
Knowledge Cell
800ca
800cb
800da
800db
800dc
800dd
800de

FIG. 28

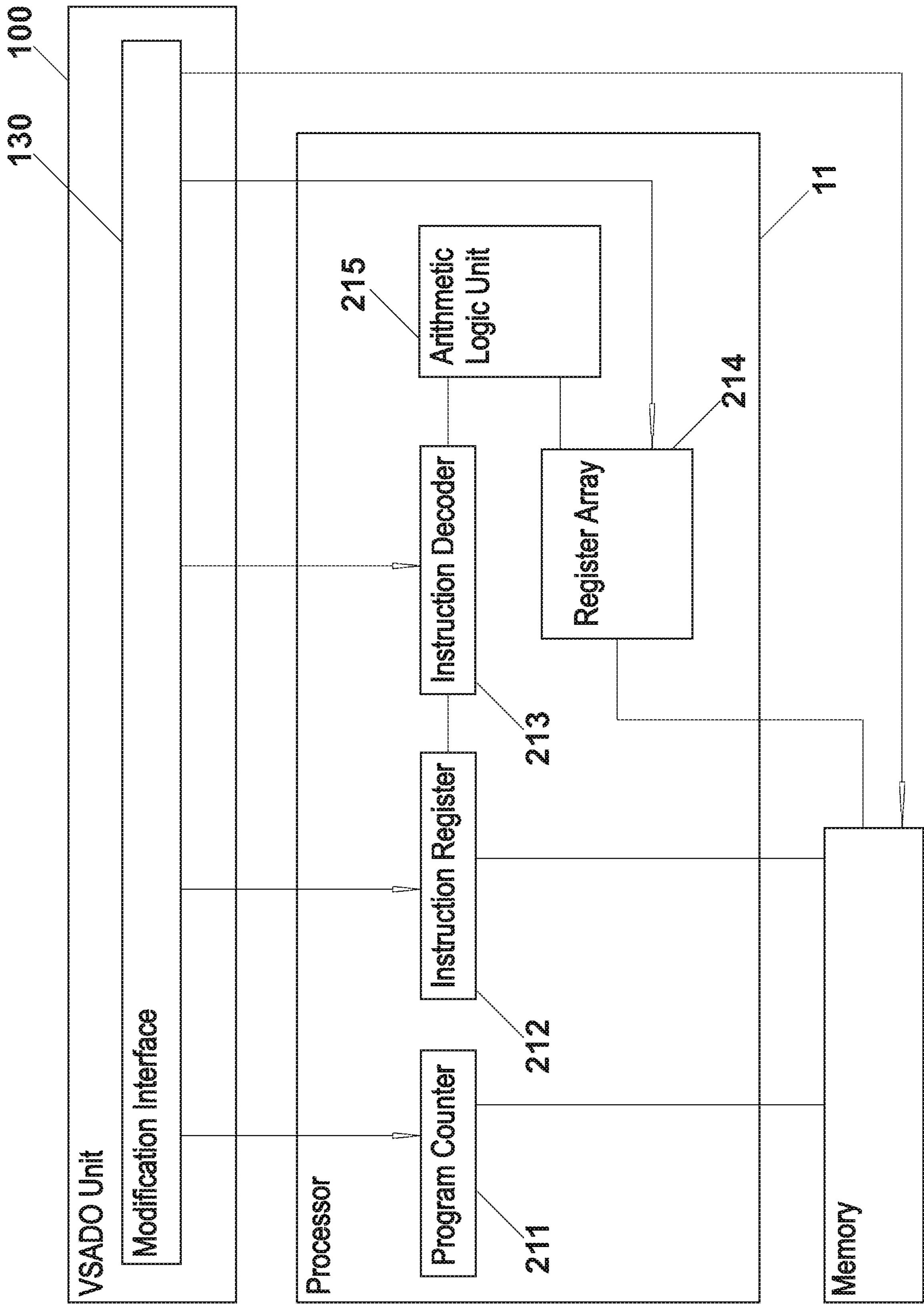

100
130
VSADO Unit
Modification Interface
Processor
11
215
Arithmetic Logic Unit
214
Instruction Decoder
Register Array
213
Instruction Register
212
Program Counter
211
Memory
12

FIG. 29

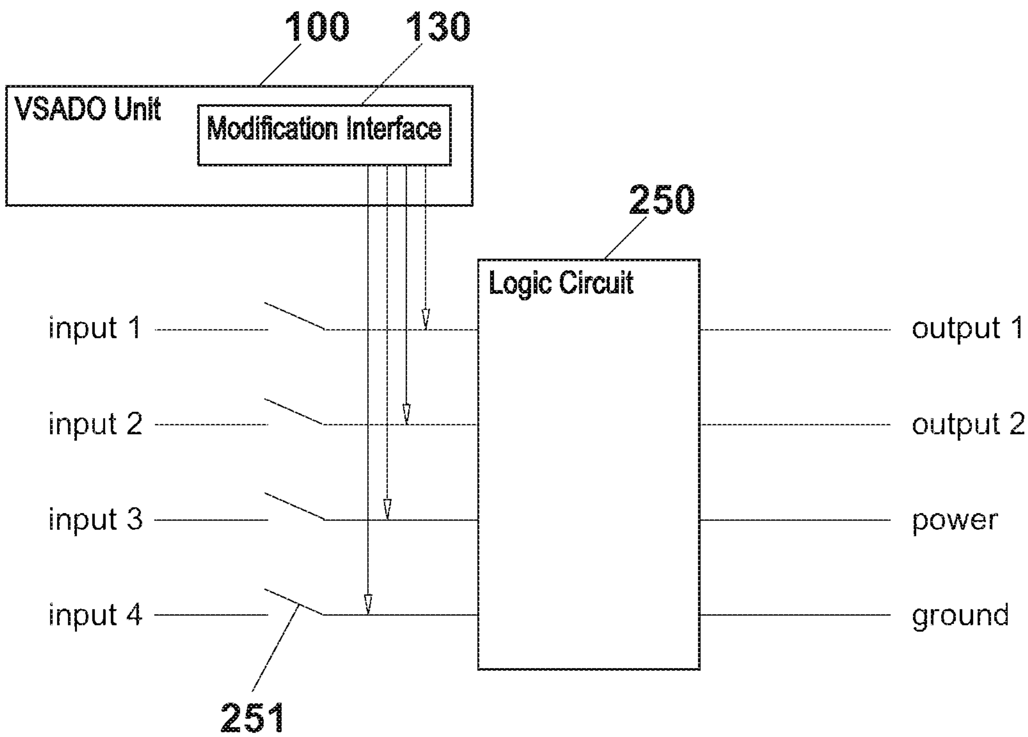


FIG. 30A

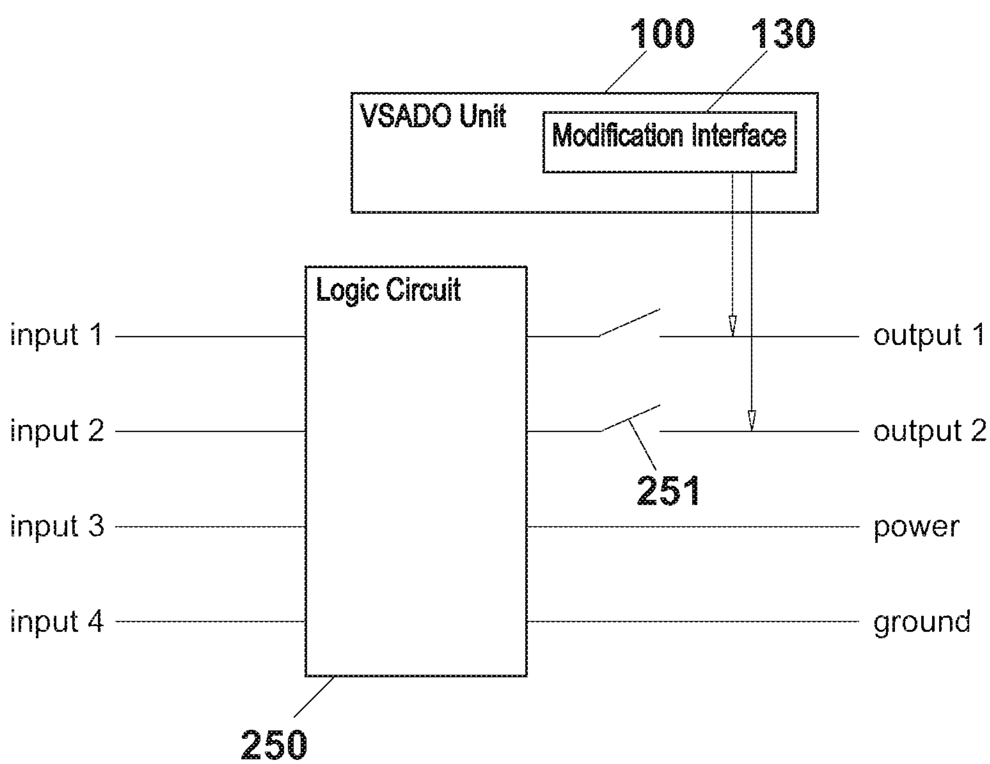


FIG. 30B

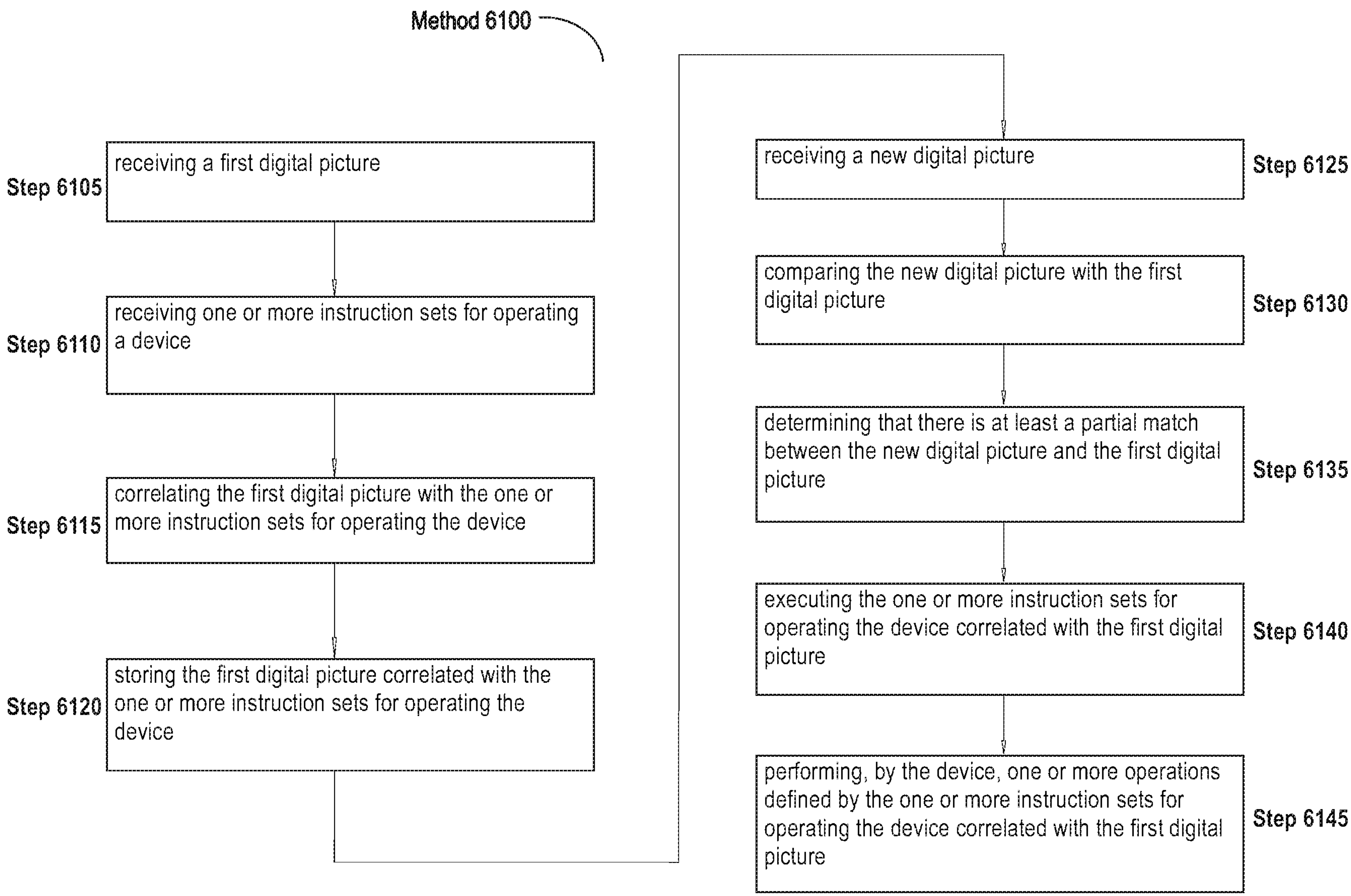

Method 6100
Step 6105
receiving a first digital picture
Step 6110
receiving one or more instruction sets for operating a device
Step 6115
correlating the first digital picture with the one or more instruction sets for operating the device
Step 6120
storing the first digital picture correlated with the one or more instruction sets for operating the device
Step 6125
receiving a new digital picture
Step 6130
comparing the new digital picture with the first digital picture
Step 6135
determining that there is at least a partial match between the new digital picture and the first digital picture
Step 6140
executing the one or more instruction sets for operating the device correlated with the first digital picture
Step 6145
performing, by the device, one or more operations defined by the one or more instruction sets for operating the device correlated with the first digital picture

FIG. 31

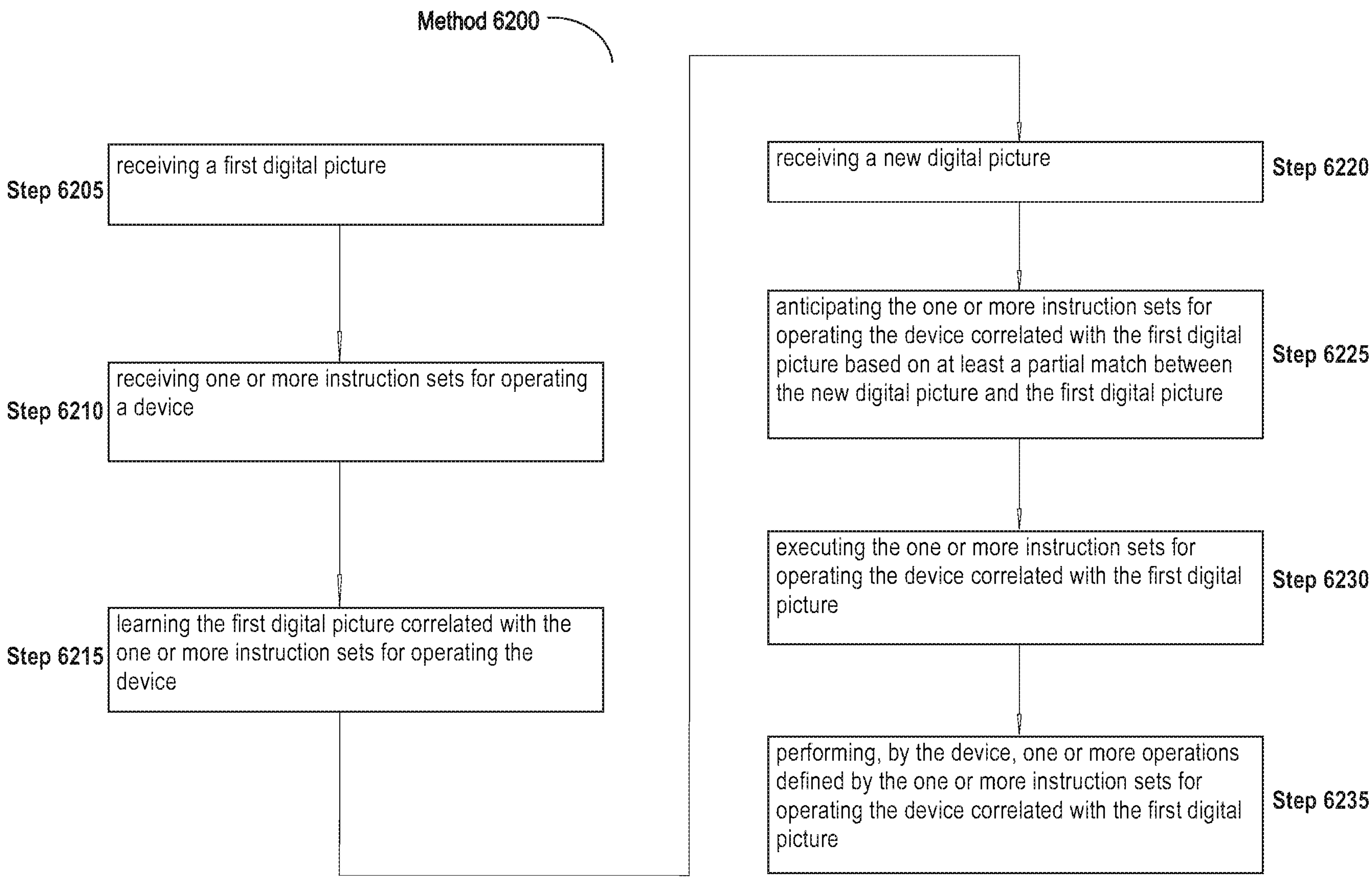

Method 6200
Step 6205
receiving a first digital picture
Step 6210
receiving one or more instruction sets for operating a device
Step 6215
learning the first digital picture correlated with the one or more instruction sets for operating the device
Step 6220
receiving a new digital picture
Step 6225
anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture
Step 6230
executing the one or more instruction sets for operating the device correlated with the first digital picture
Step 6235
performing, by the device, one or more operations defined by the one or more instruction sets for operating the device correlated with the first digital picture

FIG. 32

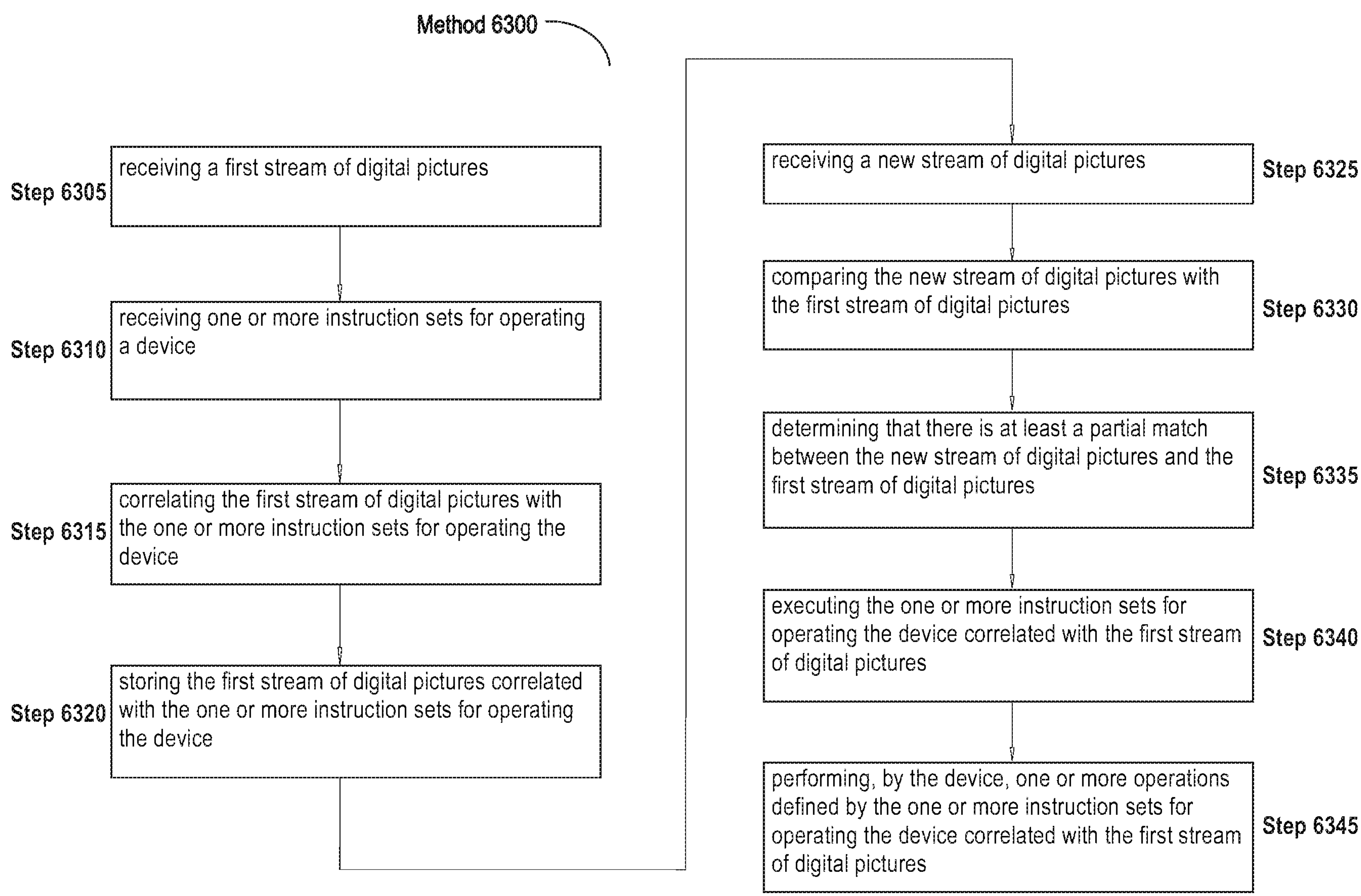

Method 6300
Step 6305
receiving a first stream of digital pictures
Step 6310
receiving one or more instruction sets for operating a device
Step 6315
correlating the first stream of digital pictures with the one or more instruction sets for operating the device
Step 6320
storing the first stream of digital pictures correlated with the one or more instruction sets for operating the device
Step 6325
receiving a new stream of digital pictures
Step 6330
comparing the new stream of digital pictures with the first stream of digital pictures
Step 6335
determining that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures
Step 6340
executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures
Step 6345
performing, by the device, one or more operations defined by the one or more instruction sets for operating the device correlated with the first stream of digital pictures

FIG. 33

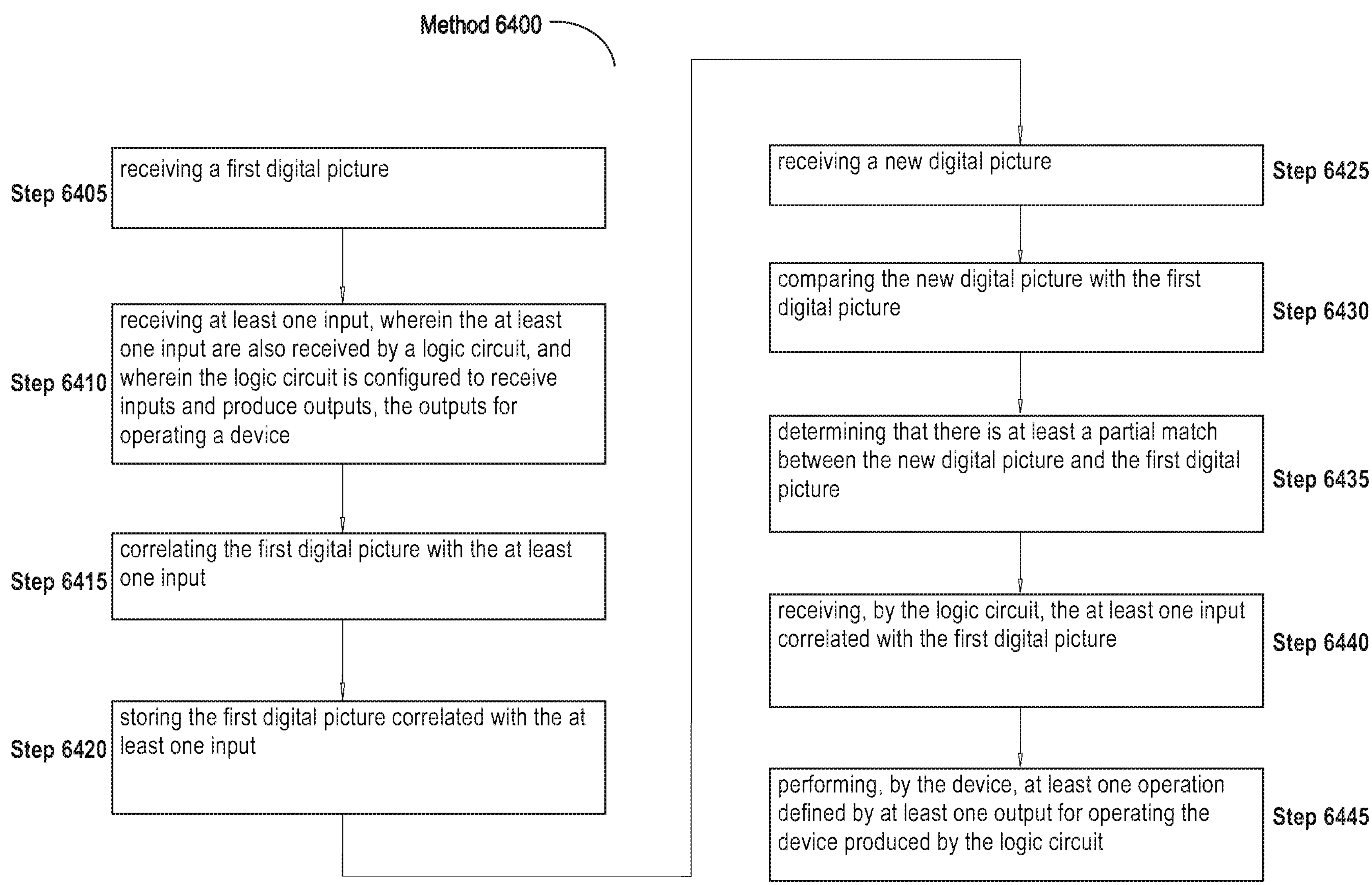

Method 6400
Step 6405
receiving a first digital picture
Step 6410
receiving at least one input, wherein the at least one input are also received by a logic circuit, and wherein the logic circuit is configured to receive inputs and produce outputs, the outputs for operating a device
Step 6415
correlating the first digital picture with the at least one input
Step 6420
storing the first digital picture correlated with the at least one input
Step 6425
receiving a new digital picture
Step 6430
comparing the new digital picture with the first digital picture
Step 6435
determining that there is at least a partial match between the new digital picture and the first digital picture
Step 6440
receiving, by the logic circuit, the at least one input correlated with the first digital picture
Step 6445
performing, by the device, at least one operation defined by at least one output for operating the device produced by the logic circuit

FIG. 34

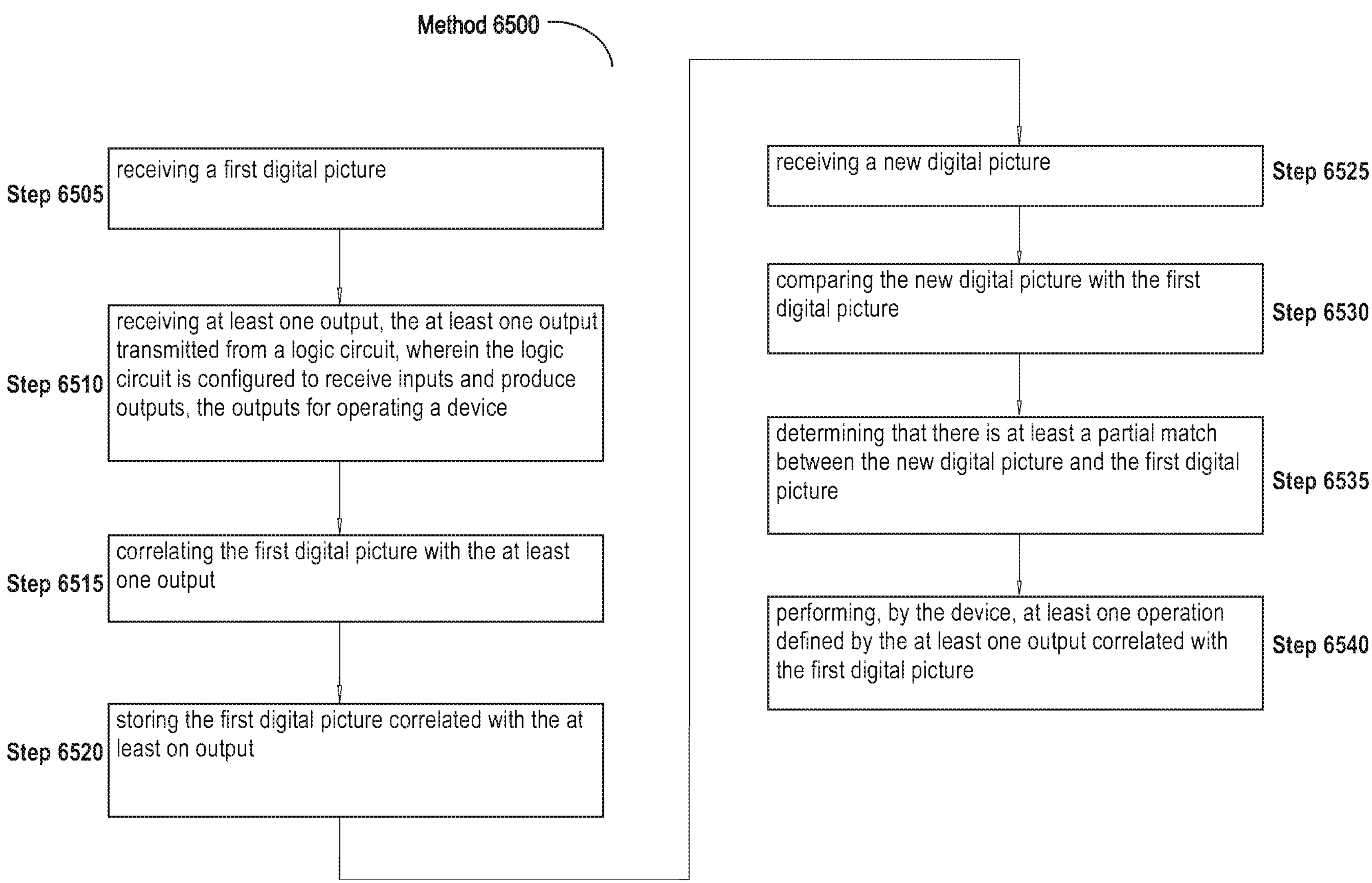

Method 6500
Step 6505
receiving a first digital picture
Step 6510
receiving at least one output, the at least one output transmitted from a logic circuit, wherein the logic circuit is configured to receive inputs and produce outputs, the outputs for operating a device
Step 6515
correlating the first digital picture with the at least one output
Step 6520
storing the first digital picture correlated with the at least on output
Step 6525
receiving a new digital picture
Step 6530
comparing the new digital picture with the first digital picture
Step 6535
determining that there is at least a partial match between the new digital picture and the first digital picture
Step 6540
performing, by the device, at least one operation defined by the at least one output correlated with the first digital picture

FIG. 35

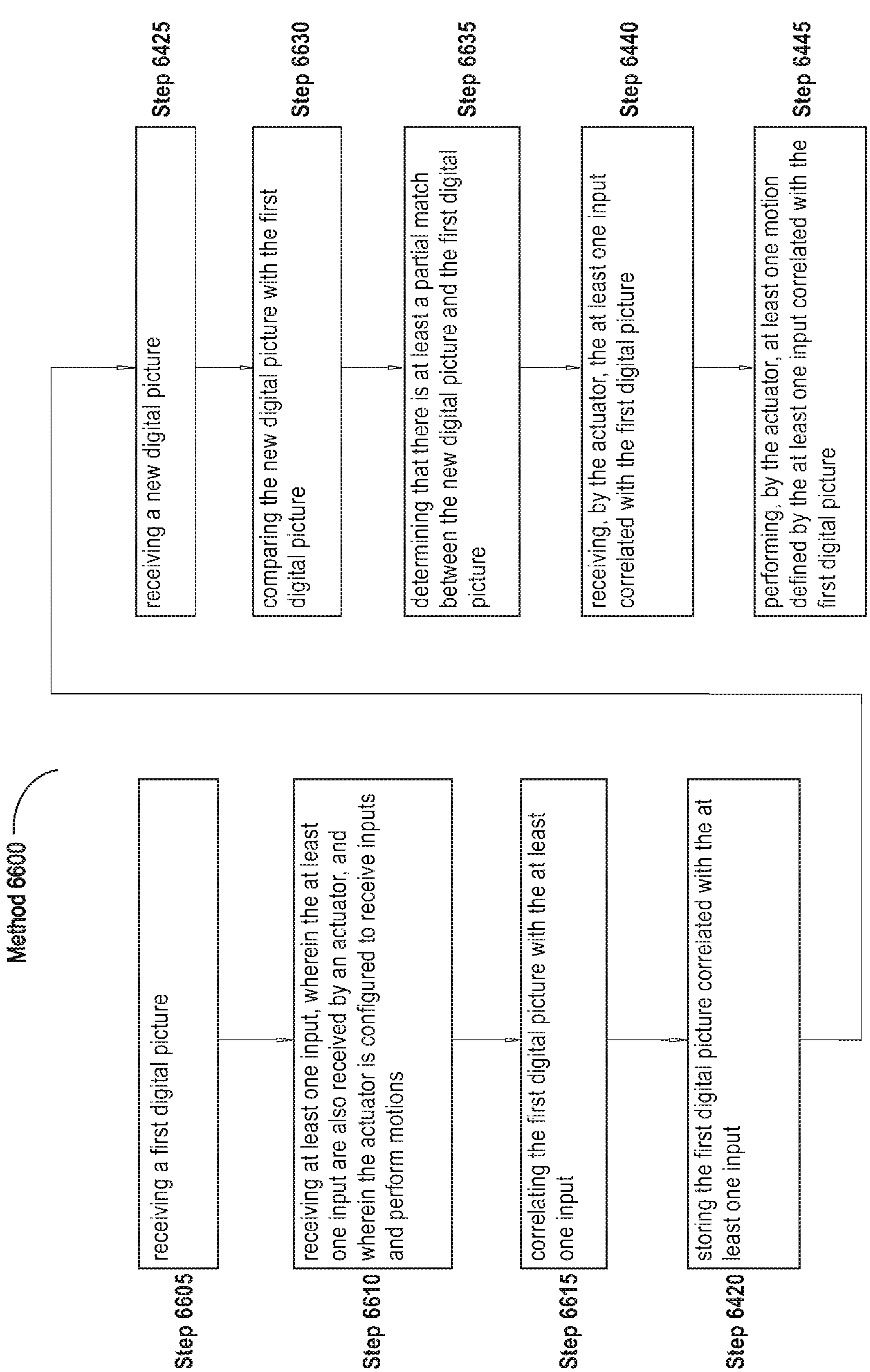

Method 6600
Step 6605
receiving a first digital picture
Step 6610
receiving at least one input, wherein the at least one input are also received by an actuator, and wherein the actuator is configured to receive inputs and perform motions
Step 6615
correlating the first digital picture with the at least one input
Step 6420
storing the first digital picture correlated with the at least one input
Step 6425
receiving a new digital picture
Step 6630
comparing the new digital picture with the first digital picture
Step 6635
determining that there is at least a partial match between the new digital picture and the first digital picture
Step 6440
receiving, by the actuator, the at least one input correlated with the first digital picture
Step 6445
performing, by the actuator, at least one motion defined by the at least one input correlated with the first digital picture

FIG. 36

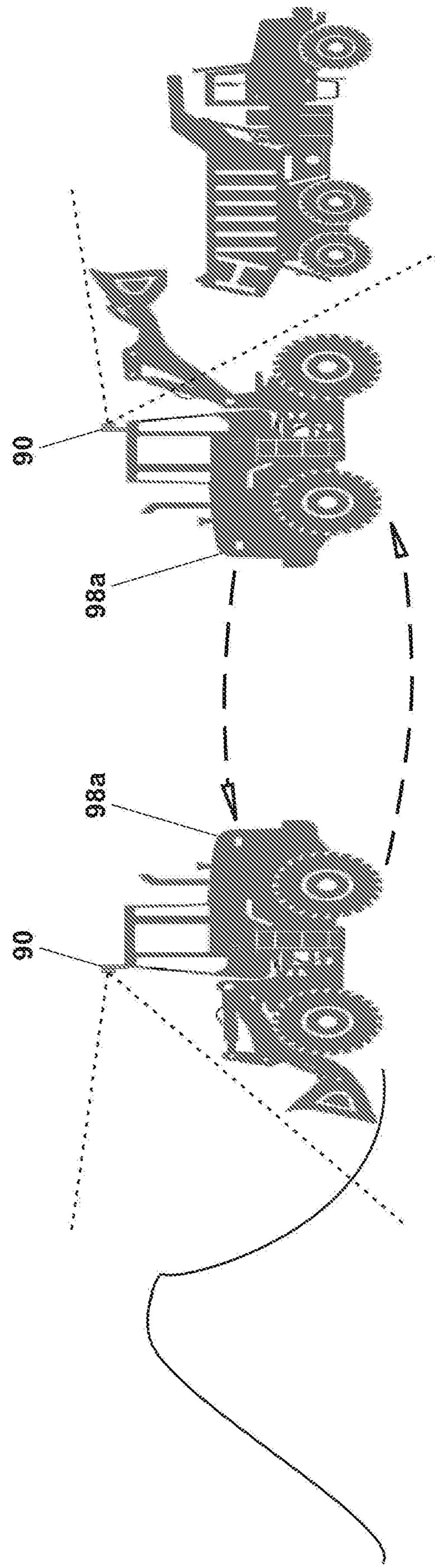


FIG. 37

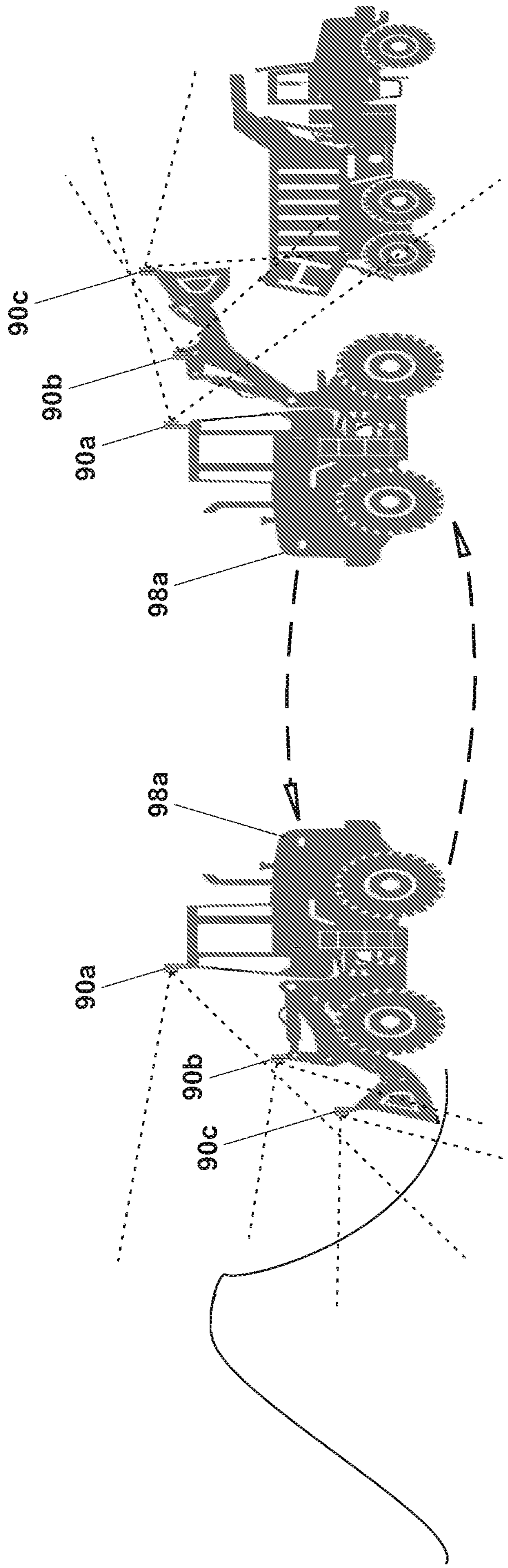


FIG. 38

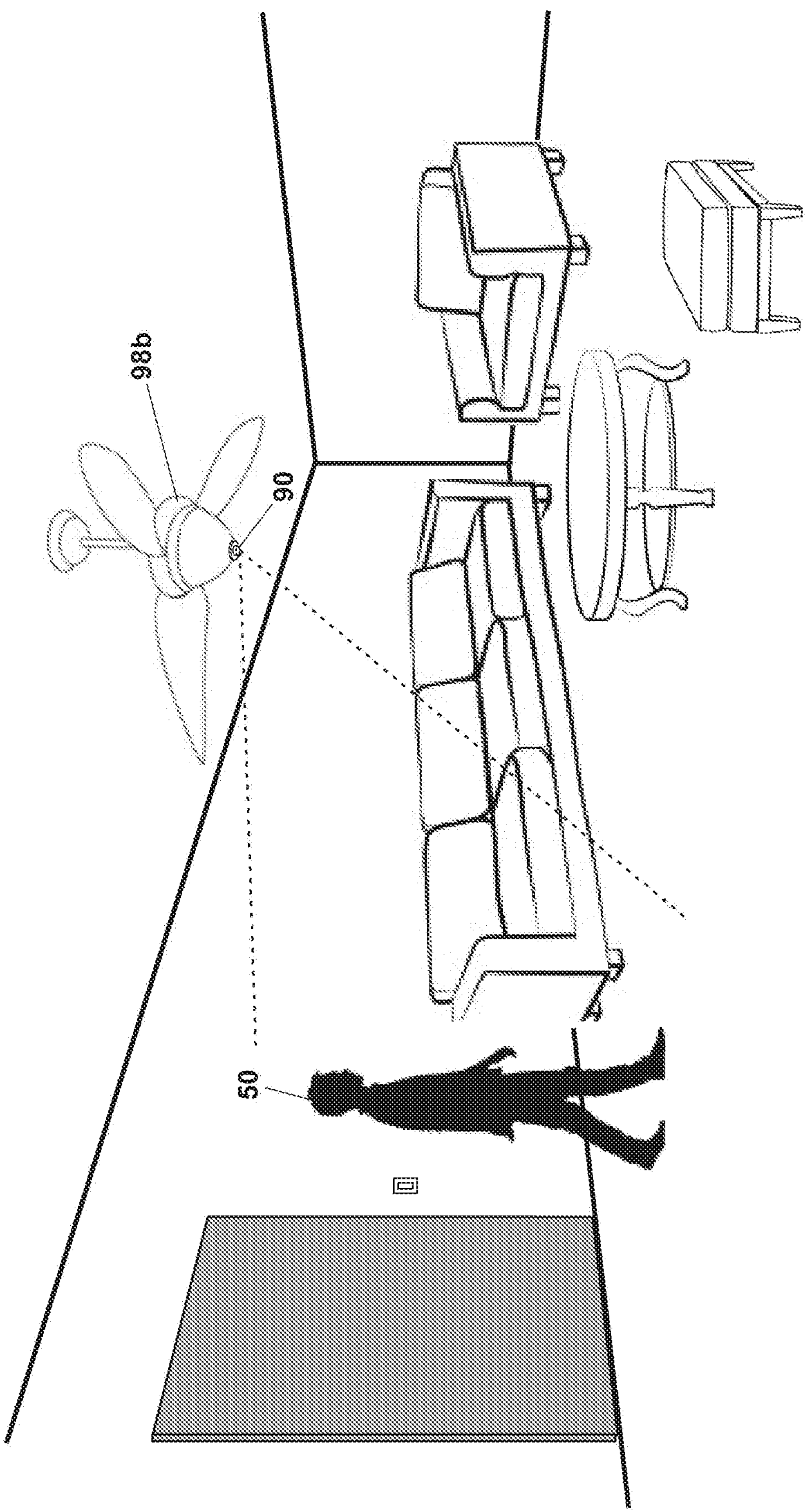


FIG. 39

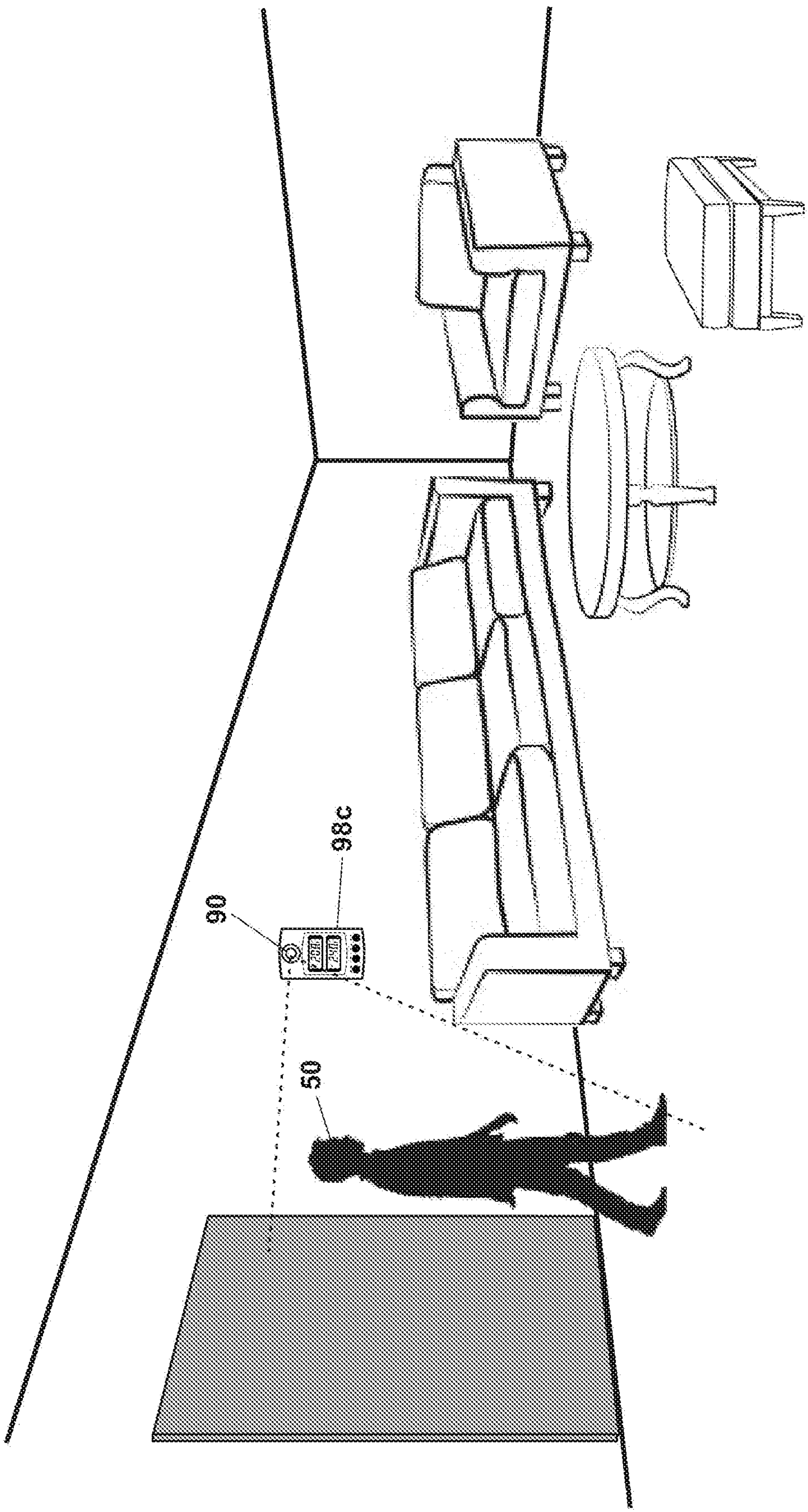


FIG. 40

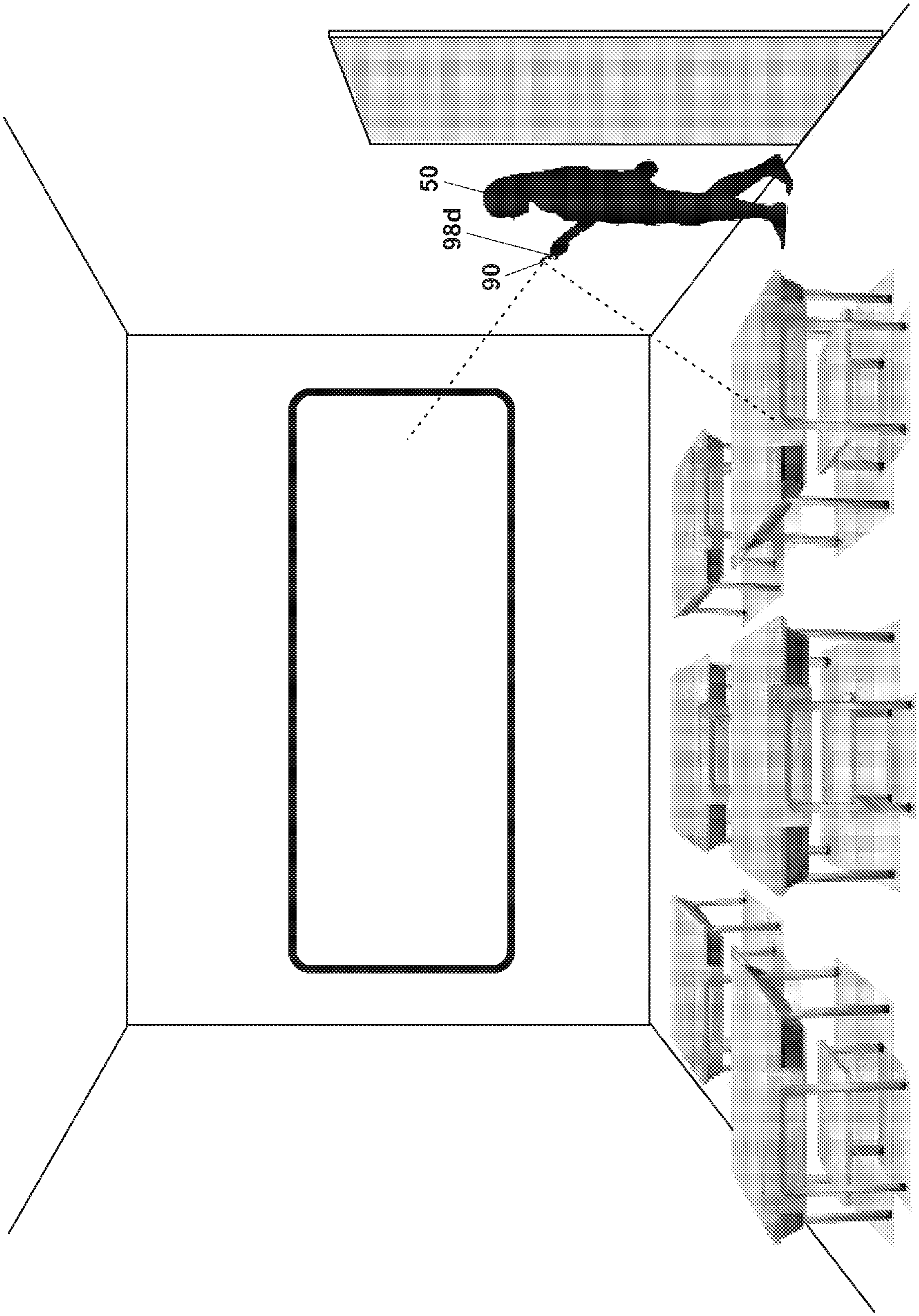


FIG. 41

MACHINE LEARNING FOR COMPUTING ENABLED SYSTEMS AND/OR DEVICES

CROSS REFERENCE TO RELATED APPLICATION

This application is a continuation of, and claims priority under 35 U.S.C. §120 from, nonprovisional U.S. patent application serial number 15/822,150 entitled “MACHINE LEARNING FOR COMPUTING ENABLED SYSTEMS AND/OR DEVICES”, filed on November 26, 2017. The disclosure of the foregoing document is incorporated herein by reference.

FIELD

The disclosure generally relates to computing enabled systems and/or devices.

COPYRIGHT NOTICE

BACKGROUND

Computing enabled systems and/or devices range from appliances, toys, entertainment electronics, computers, and communication systems and/or devices to vehicles, robots, and industrial systems and/or devices, and/or others. These systems and/or devices depend on user’s input to various degrees for their operation. A machine learning solution is needed for computing enabled systems and/or devices to be less dependent on or fully independent from user input.

SUMMARY OF THE INVENTION

In some aspects, the disclosure relates to a system. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises a processor circuit configured to execute instruction sets for operating a device. The system may further include a memory unit configured to store data. The system may further include a picture capturing apparatus configured to capture digital pictures. The system may further include an artificial intelligence unit. In some embodiments, the artificial intelligence unit may be configured to: receive a first digital picture from the picture capturing apparatus. The artificial intelligence unit may be further configured to: receive one or more instruction sets for operating the device from the processor circuit. The artificial intelligence unit may be further configured to: learn the first digital picture correlated with the one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: receive a new digital picture from the picture capturing apparatus. The artificial intelligence unit may be further configured to: anticipate the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture, the executing performed in response to the anticipating of the artificial

intelligence unit, wherein the device performs one or more operations defined by the one or more instruction sets for operating the device correlated with the first digital picture, the one or more operations performed in response to the executing by the processor circuit.

In certain embodiments, at least one of the processor circuit, the memory unit, the picture capturing apparatus, or the artificial intelligence unit are part of, operating on, or coupled to the device. In further embodiments, the device includes one or more devices. In further embodiments, the device includes a smartphone, a fixture, a control device, a computing enabled device, or a computer.

In some embodiments, the processor circuit includes one or more processor circuits. In further embodiments, the processor circuit includes a logic circuit. The logic circuit may include a microcontroller. The one or more instruction sets may include one or more inputs into or one or more outputs from the logic circuit.

In certain embodiments, the processor circuit includes a logic circuit, the instruction sets for operating the device include inputs into the logic circuit, and executing instruction sets for operating the device includes performing logic operations on the inputs into the logic circuit and producing outputs for operating the device. The logic circuit may include a microcontroller. In further embodiments, the processor circuit includes a logic circuit, the instruction sets for operating the device include outputs from the logic circuit for operating the device, and executing instruction sets for operating the device includes performing logic operations on inputs into the logic circuit and producing the outputs from the logic circuit for operating the device.

In some embodiments, the memory unit includes one or more memory units. In further embodiments, the memory unit resides on a remote computing device, the remote computing device coupled to the processor circuit via a network. The remote computing device may include a server.

In some embodiments, the picture capturing apparatus includes one or more picture capturing apparatuses. In further embodiments, the picture capturing apparatus includes a motion picture camera or a still picture camera. In further embodiments, the picture capturing apparatus resides on a remote device, the remote device coupled to the processor circuit via a network.

In certain embodiments, the artificial intelligence unit is coupled to the picture capturing apparatus. In further embodiments, the artificial intelligence unit is coupled to the memory unit. In further embodiments, the artificial intelligence unit is part of, operating on, or coupled to the processor circuit. In further embodiments, the system further comprises: a second processor circuit, wherein the artificial intelligence unit is part of, operating on, or coupled to the second processor circuit. In further embodiments, the artificial intelligence unit is part of, operating on, or coupled to a remote computing device, the remote computing device coupled to the processor circuit via a network. In further embodiments, the artificial intelligence unit includes a circuit, a computing apparatus, or a computing system attachable to the processor circuit. In further embodiments, the artificial intelligence unit includes a circuit, a computing apparatus, or a computing system attachable to the device. In further embodiments, the artificial intelligence unit is attachable to an application for operating the device, the application running on the processor circuit. In further embodiments, the artificial intelligence unit includes a circuit, a computing apparatus, or a computing system built into the processor circuit. In further embodiments, the artificial intelligence unit includes a circuit, a computing apparatus, or a computing system built into the device. In further embodiments, the artificial intelligence unit is built into an application for operating the device, the application running on the processor circuit. In further embodiments, the artificial intelligence unit is provided as a feature of the processor circuit. In further

embodiments, the artificial intelligence unit is provided as a feature of an application running on the processor circuit. In further embodiments, the artificial intelligence unit is provided as a feature of the device. In further embodiments, the artificial intelligence unit is further configured to: take control from, share control with, or release control to the processor circuit. In further embodiments, the artificial intelligence unit is further configured to: take control from, share control with, or release control to an application or an object of the application, the application running on the processor circuit. In further embodiments, the artificial intelligence unit is further configured to: take control from, share control with, or release control to a user or a system.

In some embodiments, the first digital picture includes a stream of digital pictures. In further embodiments, the new digital picture includes a stream of digital pictures. In further embodiments, the first and the new digital pictures portray the device's surrounding. In further embodiments, the first and the new digital pictures portray a remote device's surrounding. In further embodiments, the first or the new digital picture includes a JPEG picture, a GIF picture, a TIFF picture, a PNG picture, a PDF picture, or a digitally encoded picture. The stream of digital pictures may include a MPEG motion picture, an AVI motion picture, a FLV motion picture, a MOV motion picture, a RM motion picture, a SWF motion picture, a WMV motion picture, a DivX motion picture, or a digitally encoded motion picture. In further embodiments, the first digital picture includes a comparative digital picture whose at least one portion can be used for comparisons with at least one portion of digital pictures subsequent to the first digital picture, the digital pictures subsequent to the first digital picture comprising the new digital picture. In further embodiments, the first digital picture includes a comparative digital picture that can be used for comparisons with the new digital picture. In further embodiments, the new digital picture includes an anticipatory digital picture whose correlated one or more instruction sets can be used for anticipation of one or more instruction sets to be executed by the processor circuit.

In certain embodiments, the one or more instruction sets for operating the device include one or more instruction sets that temporally correspond to the first digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed at a time of the capturing the first digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed prior to the capturing the first digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed within a threshold period of time prior to the capturing the first digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed subsequent to the capturing the first digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed within a threshold period of time subsequent to the capturing the first digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed within a threshold period of time prior to the capturing the first digital picture or a threshold period of time subsequent to the capturing the first digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed from a start of capturing a preceding digital picture to a start of capturing the first digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed from a start of capturing the first digital picture to a start of capturing a subsequent digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed from a

completion of capturing a preceding digital picture to a completion of capturing the first digital picture. The one or more instruction sets that temporally correspond to the first digital picture include one or more instruction sets executed from a completion of capturing the first digital picture to a completion of capturing a subsequent digital picture.

In some embodiments, the one or more instruction sets for operating the device are executed by the processor circuit. In further embodiments, the one or more instruction sets for operating the device are part of an application for operating the device, the application running on the processor circuit. In further embodiments, the one or more instruction sets for operating the device include one or more inputs into or one or more outputs from the processor circuit. In further embodiments, the one or more instruction sets for operating the device include values or states of one or more registers or elements of the processor circuit. In further embodiments, an instruction set includes at least one of: a command, a keyword, a symbol, an instruction, an operator, a variable, a value, an object, a data structure, a function, a parameter, a state, a signal, an input, an output, a character, a digit, or a reference thereto. In further embodiments, the one or more instruction sets include a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, a structured query language (SQL) code, or a machine code. In further embodiments, the one or more instruction sets include one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the processor circuit includes a logic circuit. The one or more instruction sets for operating the device include one or more inputs into a logic circuit. The one or more instruction sets for operating the device include one or more outputs from a logic circuit. In further embodiments, the one or more instruction sets for operating the device include one or more instruction sets for operating an application or an object of the application, the application running on the processor circuit.

In some embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes obtaining the one or more instruction sets from the processor circuit. In further embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes receiving the one or more instruction sets as they are executed by the processor circuit. In further embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes receiving the one or more instruction sets for operating the device from a register or an element of the processor circuit. In further embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes receiving the one or more instruction sets for operating the device from an element that is part of, operating on, or coupled to the processor circuit. In further embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes receiving the one or more instruction sets for operating the device from at least one of: the memory unit, the device, a virtual machine, a runtime engine, a hard drive, a storage device, a peripheral device, a network connected device, or a user. In further embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes receiving the one or more instruction sets from a plurality of processor circuits, applications, memory units, devices, virtual machines, runtime engines, hard drives, storage devices, peripheral devices, network connected devices, or users.

In certain embodiments, the processor circuit includes a logic circuit, and wherein the receiving the one or more instruction sets for operating the device from the processor circuit includes receiving the one or more

instruction sets for operating the device from the logic circuit. The logic circuit may include a microcontroller. The receiving the one or more instruction sets for operating the device from the logic circuit may include receiving the one or more instruction sets for operating the device from an element of the logic circuit. The receiving the one or more instruction sets for operating the device from the logic circuit may include receiving one or more inputs into the logic circuit. The receiving the one or more instruction sets for operating the device from the logic circuit may include receiving one or more outputs from the logic circuit.

In some embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes receiving the one or more instruction sets for operating the device from an application for operating the device, the application running on the processor circuit. In further embodiments, the system further comprises: an application including instruction sets for operating the device, the application running on the processor circuit, wherein the receiving the one or more instruction sets for operating the device from the processor circuit includes receiving the one or more instruction sets for operating the device from the application.

In certain embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes receiving the one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes at least one of: tracing, profiling, or instrumentation of a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, a structured query language (SQL) code, or a machine code. In further embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes at least one of: tracing, profiling, or instrumentation of an element that is part of, operating on, or coupled to the processor circuit. In further embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes at least one of: tracing, profiling, or instrumentation of a register of the processor circuit, the memory unit, a storage, or a repository where the one or more instruction sets for operating the device are stored. In further embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes at least one of: tracing, profiling, or instrumentation of the processor circuit, the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a processing element. In further embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes at least one of: tracing, profiling, or instrumentation of the processor circuit or tracing, profiling, or instrumentation of a component of the processor circuit. In further embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes at least one of: tracing, profiling, or instrumentation of an application or an object of the application, the application running on the processor circuit. In further embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes at least one of: tracing, profiling, or instrumentation at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes at least one of: tracing, profiling, or instrumentation of one or more of code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes at least one of: tracing, profiling, or instrumentation of a user input. In further embodiments, the receiving the one or more instruction

sets for operating the device from the processor circuit includes at least one of: a manual, an automatic, a dynamic, or a just in time (JIT) tracing, profiling, or instrumentation. In further embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes utilizing at least one of: a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, a logging tool, or an independent tool for obtaining instruction sets. In further embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes utilizing an assembly language. In further embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes utilizing a branch or a jump. In further embodiments, the receiving the one or more instruction sets for operating the device from the processor circuit includes a branch tracing or a simulation tracing.

In some embodiments, the system further comprises: an interface configured to receive instruction sets, wherein the one or more instruction sets for operating the device are received by the interface. The interface may include an acquisition interface.

In certain embodiments, the first digital picture correlated with the one or more instruction sets for operating the device includes a unit of knowledge of how the device operated in a visual surrounding. In further embodiments, the first digital picture correlated with the one or more instruction sets for operating the device is included in a neuron, a node, a vertex, or an element of a data structure. The data structure may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, or a knowledge structure. Some of the neurons, nodes, vertices, or elements may be interconnected.

In some embodiments, the first digital picture correlated with the one or more instruction sets for operating the device is structured into a knowledge cell. In further embodiments, the knowledge cell includes a unit of knowledge of how the device operated in a visual surrounding. In further embodiments, the knowledge cell is included in a neuron, a node, a vertex, or an element of a data structure. The data structure may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, or a knowledge structure. Some of the neurons, nodes, vertices, or elements may be interconnected.

In certain embodiments, the learning the first digital picture correlated with the one or more instruction sets for operating the device includes correlating the first digital picture with the one or more instruction sets for operating the device. The correlating the first digital picture with the one or more instruction sets for operating the device may include generating a knowledge cell, the knowledge cell comprising the first digital picture correlated with the one or more instruction sets for operating the device. The correlating the first digital picture with the one or more instruction sets for operating the device may include structuring a unit of knowledge of how the device operated in a visual surrounding.

In some embodiments, the learning the first digital picture correlated with the one or more instruction sets for operating the device includes learning a user's knowledge, style, or methodology of operating the device in a visual surrounding. In further embodiments, the learning the first digital picture correlated with the one or more instruction sets for operating the device includes spontaneous learning the first digital picture correlated with the one or more instruction sets for operating the device.

In some embodiments, the learning the first digital picture correlated with the one or more instruction sets for operating the device includes storing, into the memory unit, the first digital picture correlated with the one or more instruction sets for operating the device, the first digital picture correlated with the one or more instruction sets for

operating the device being part of a stored plurality of digital pictures correlated with one or more instruction sets for operating the device. In further embodiments, the plurality of digital pictures correlated with one or more instruction sets for operating the device include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, the plurality of digital pictures correlated with one or more instruction sets for operating the device are organized into a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, each of the plurality of digital pictures correlated with one or more instruction sets for operating the device is included in a neuron, a node, a vertex, or an element of a data structure. The data structure may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, or a knowledge structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the plurality of digital pictures correlated with one or more instruction sets for operating the device include a user's knowledge, style, or methodology of operating the device in visual surroundings. In further embodiments, the plurality of digital pictures correlated with one or more instruction sets for operating the device are stored on a remote computing device. In further embodiments, the plurality of digital pictures correlated with one or more instruction sets for operating the device include an artificial intelligence system for knowledge structuring, storing, or representation. The artificial intelligence system for knowledge structuring, storing, or representation may include at least one of: a deep learning system, a supervised learning system, an unsupervised learning system, a neural network, a search-based system, an optimization-based system, a logic-based system, a fuzzy logic-based system, a tree-based system, a graph-based system, a hierarchical system, a symbolic system, a sub-symbolic system, an evolutionary system, a genetic system, a multi-agent system, a deterministic system, a probabilistic system, or a statistical system.

In certain embodiments, the anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture includes comparing at least one portion of the new digital picture with at least one portion of the first digital picture. The at least one portion of the new digital picture may include at least one region, at least one feature, or at least one pixel of the new digital picture. The at least one portion of the first digital picture may include at least one region, at least one feature, or at least one pixel of the first digital picture. The comparing the at least one portion of the new digital picture with the at least one portion of the first digital picture may include comparing at least one region of the new digital picture with at least one region of the first digital picture. The comparing the at least one portion of the new digital picture with the at least one portion of the first digital picture may include comparing at least one feature of the new digital picture with at least one feature of the first digital picture. The at least one portion of the new digital picture with the at least one portion of the first digital picture may include comparing at least one pixel of the new digital picture with at least one pixel of the first digital picture. The comparing the at least one portion of the new digital picture with the at least one portion of the first digital picture may include at least one of: performing a color adjustment, performing a size adjustment, performing a content manipulation, utilizing a transparency, or utilizing a mask on the new or the first digital picture. The comparing the at least one portion of the new digital picture with the at least one portion of the first digital picture may include recognizing at least one person or object in the new digital picture and at least one person or object in the first digital picture, and comparing the at least one person or object from the new digital picture with the at least one person or object from the first digital picture.

In some embodiments, he anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture includes determining that there is at least a partial match between the new digital picture and the first digital picture. In further embodiments, the determining that there is at least a partial match between the new digital picture and the first digital picture includes determining that there is at least a partial match between one or more portions of the new digital picture and one or more portions of the first digital picture. In further embodiments, the determining that there is at least a partial match between the new digital picture and the first digital picture includes determining that a similarity between at least one portion of the new digital picture and at least one portion of the first digital picture exceeds a similarity threshold. In further embodiments, the determining that there is at least a partial match between the new digital picture and the first digital picture includes determining a substantial similarity between at least one portion of the new digital picture and at least one portion of the first digital picture. The at least one portion of the new digital picture may include at least one region, at least one feature, or at least one pixel of the new digital picture. The at least one portion of the first digital picture may include at least one region, at least one feature, or at least one pixel of the first digital picture. The substantial similarity may be achieved when a similarity between the at least one portion of the new digital picture and the at least one portion of the first digital picture exceeds a similarity threshold. The substantial similarity may be achieved when a number or a percentage of matching or partially matching regions from the new digital picture and from the first digital picture exceeds a threshold number or threshold percentage. The substantial similarity may be achieved when a number or a percentage of matching or partially matching features from the new digital picture and from the first digital picture exceeds a threshold number or threshold percentage. The substantial similarity may be achieved when a number or a percentage of matching or partially matching pixels from the new digital picture and from the first digital picture exceeds a threshold number or threshold percentage. The substantial similarity may be achieved when one or more same or similar objects are recognized in the new digital picture and the first digital picture. In further embodiments, the determining that there is at least a partial match between the new digital picture and the first digital picture includes determining that a number or a percentage of matching regions from the new digital picture and from the first digital picture exceeds a threshold number or threshold percentage. The matching regions from the new digital picture and from the first digital picture may be determined factoring in at least one of: a location of a region, an importance of a region, a threshold for a similarity in a region, or a threshold for a difference in a region. In further embodiments, the determining that there is at least a partial match between the new digital picture and the first digital picture includes determining that a number or a percentage of matching features from the new digital picture and from the first digital picture exceeds a threshold number or threshold percentage. The matching features from the new digital picture and from the first digital picture may be determined factoring in at least one of: a type of a feature, an importance of a feature, a location of a feature, a threshold for a similarity in a feature, or a threshold for a difference in a feature. In further embodiments, the determining that there is at least a partial match between the new digital picture and the first digital picture includes determining that a number or a percentage of matching pixels from the new digital picture and from the first digital picture exceeds a threshold number or threshold percentage. The matching pixels from the new digital picture and from the first digital picture may be determined factoring in at least one of: a location of a pixel, a threshold for a similarity in a pixel, or a threshold for a difference in a pixel. In further embodiments, the

determining that there is at least a partial match between the new digital picture and the first digital picture includes recognizing a same person or object in the new and the first digital pictures.

In some embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture instead of or prior to an instruction set that would have been executed next. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes modifying one or more instruction sets of the processor circuit. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes modifying a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes inserting the one or more instruction sets for operating the device correlated with the first digital picture into a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes redirecting the processor circuit to the one or more instruction sets for operating the device correlated with the first digital picture. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes redirecting the processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the one or more instruction sets for operating the device correlated with the first digital picture. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes transmitting, to the processor circuit for execution, the one or more instruction sets for operating the device correlated with the first digital picture. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes issuing an interrupt to the processor circuit and executing the one or more instruction sets for operating the device correlated with the first digital picture following the interrupt. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes modifying an element that is part of, operating on, or coupled to the processor circuit.

In certain embodiments, the processor circuit includes a logic circuit, and wherein the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes causing the logic circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture. The logic circuit may include a microcontroller. The causing the logic circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture may include modifying an element of the logic circuit. The causing the logic circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture may include inserting the one or more instruction sets for operating the device correlated with the first digital picture into an element of the logic circuit. The causing the logic circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture may include redirecting the logic circuit to the one or more instruction sets for operating the device correlated with the first digital picture. The causing the logic circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture may include replacing inputs into the logic circuit with the one or more instruction sets for

operating the device correlated with the first digital picture. The causing the logic circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture may include replacing outputs from the logic circuit with the one or more instruction sets for operating the device correlated with the first digital picture.

In certain embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes causing an application for operating the device to execute the one or more instruction sets for operating the device correlated with the first digital picture, the application running on the processor circuit.

In further embodiments, the system further comprises: an application including instruction sets for operating the device, the application running on the processor circuit, wherein the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes modifying the application.

In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes redirecting an application to the one or more instruction sets for operating the device correlated with the first digital picture, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes redirecting an application to one or more alternate instruction sets, the application running on the processor circuit, the alternate instruction sets comprising the one or more instruction sets for operating the device correlated with the first digital picture. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes modifying one or more instruction sets of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes modifying at least one of: the memory unit, a register of the processor circuit, a storage, or a repository where instruction sets are stored or used. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes modifying one or more instruction sets for operating an application or an object of the application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes modifying at least one of: an element of the processor circuit, an element of the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes modifying one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device

correlated with the first digital picture includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes utilizing an assembly language. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes adding or inserting additional code into a code of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes at least one of: modifying, removing, rewriting, or overwriting a code of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture includes at least one of: branching, redirecting, extending, or hot swapping a code of an application, the application running on the processor circuit. The branching or redirecting the code may include inserting at least one of: a branch, a jump, or a means for redirecting an execution. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes implementing a user's knowledge, style, or methodology of operating the device in a visual surrounding.

In some embodiments, the system further comprises: an interface configured to cause execution of instruction sets, wherein the executing the one or more instruction sets for operating the device correlated with the first digital picture is caused by the interface. The interface may include a modification interface.

In certain embodiments, the one or more operations defined by the one or more instruction sets for operating the device correlated with the first digital picture include at least one of: an operation with or by a smartphone, an operation with or by a fixture, an operation with or by a control device, or an operation with or by a computer or computing enabled device.

In some embodiments, the performing the one or more operations defined by the one or more instruction sets for operating the device correlated with the first digital picture includes implementing a user's knowledge, style, or methodology of operating the device in a visual surrounding.

In certain embodiments, the system further comprises: an application running on the processor circuit.

In some embodiments, the instruction sets for operating the device are part of an application for operating the device, the application running on the processor circuit.

In certain embodiments, the system further comprises: an application for operating the device, the application running on the processor circuit. The application for operating the device may include the instruction sets for operating the device.

In some embodiments, the artificial intelligence unit is further configured to: receive at least one extra information. In further embodiments, the at least one extra information include one or more of: a time information, a location information, a computed information, an observed information, a sensory information, or a contextual information. In further embodiments, the at least one extra information include one or more of: an information on a digital picture, an information on an object in the digital picture, an information on the device's visual surrounding, an information on an instruction set, an information on an application, an information on an object of the application, an information on the processor circuit, an information on the device, or an information on an user. In further embodiments, the artificial intelligence unit is further configured to: learn the first digital picture correlated with the at least one extra information. The learning the first digital picture correlated with at least one extra information may include correlating the first digital picture with the at least one extra information. The learning the first digital picture correlated with at least one extra information may include storing the first digital picture correlated with the at least one extra information into the memory unit. In further embodiments, the anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture includes anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between an extra information correlated with the new digital picture and an extra information correlated with the first digital picture. The anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between an extra information correlated with the new digital picture and an extra information correlated with the first digital picture may include comparing an extra information correlated with the new digital picture and an extra information correlated with the first digital picture. The anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between an extra information correlated with the new digital picture and an extra information correlated with the first digital picture may include determining that a similarity between an extra information correlated with the new digital picture and an extra information correlated with the first digital picture exceeds a similarity threshold.

In some embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: present, via the user interface, a user with an option to execute the one or more instruction sets for operating the device correlated with the first digital picture.

In certain embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: receive, via the user interface, a user's selection to execute the one or more instruction sets for operating the device correlated with the first digital picture.

In some embodiments, the artificial intelligence unit is further configured to: rate the executed one or more instruction sets for operating the device correlated with the first digital picture. The rating the executed one or more instruction sets for operating the device correlated with the first digital picture may include displaying, on a display, the executed one or more instruction sets for operating the device correlated with the first digital picture along with one or more rating values as options to be selected by a user. The rating the executed one or more instruction sets for operating the device correlated with the first digital picture may include rating the executed one or more instruction sets for operating the device correlated with the first digital picture without a user input. The rating the executed one or more instruction sets for operating the device correlated with the first digital picture may include associating one or more rating values with the executed one or more instruction sets for operating the device correlated with the first digital picture and storing the one or more rating values into the memory unit.

In certain embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: present, via the user interface, a user with an option to cancel the execution of the executed one or more instruction sets for operating the device correlated with the first digital picture. In further embodiments, the canceling the execution of the executed one or more instruction sets for operating the device correlated with the first digital picture includes restoring the processor circuit or the device to a prior state. The restoring the processor circuit or the device to a prior state may include saving the state of the processor circuit or the device prior to executing the one or more instruction sets for operating the device correlated with the first digital picture.

In some embodiments, the system further comprises: an input device configured to receive a user's operating directions, the user's operating directions for instructing the processor circuit on how to operate the device.

In certain embodiments, the autonomous device operating includes a partially or a fully autonomous device operating. The partially autonomous device operating may include executing the one or more instruction sets for operating the device correlated with the first digital picture responsive to a user confirmation. The fully autonomous device operating may include executing the one or more instruction sets for operating the device correlated with the first digital picture without a user confirmation.

In some embodiments, the artificial intelligence unit is further configured to: receive a second digital picture from the picture capturing apparatus; receive additional one or more instruction sets for operating the device from the processor circuit; and learn the second digital picture correlated with the additional one or more instruction sets for operating the device. In further embodiments, the second digital picture includes a second stream of digital pictures. In further embodiments, the learning the first digital picture correlated with the one or more instruction sets for operating the device and the learning the second digital picture correlated with the additional one or more instruction sets for operating the device include creating a connection between the first digital picture correlated with the one or more instruction sets for operating the device and the second digital picture correlated with the additional one or more instruction sets for operating the device. The connection may include or is associated with at least one of: an occurrence count, a weight, a parameter, or a data. In further embodiments, the learning the first digital picture correlated with the one or more instruction sets for operating the device and the learning the second digital picture correlated with the additional one or more instruction sets for operating the device include updating a connection between the first digital picture correlated with the one or more instruction sets for operating the device and the second digital picture correlated with the additional one or more instruction sets for operating the device. The

updating the connection between the first digital picture correlated with the one or more instruction sets for operating the device and the second digital picture correlated with the additional one or more instruction sets for operating the device may include updating at least one of: an occurrence count, a weight, a parameter, or a data included in or associated with the connection. In further embodiments, the learning the first digital picture correlated with the one or more instruction sets for operating the device includes storing the first digital picture correlated with the one or more instruction sets for operating the device into a first node of a data structure, and wherein the learning the second digital picture correlated with the additional one or more instruction sets for operating the device includes storing the second digital picture correlated with the additional one or more instruction sets for operating the device into a second node of the data structure. The data structure may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, or a knowledge structure. The learning the first digital picture correlated with the one or more instruction sets for operating the device and the learning the second digital picture correlated with the additional one or more instruction sets for operating the device may include creating a connection between the first node and the second node. The learning the first digital picture correlated with the one or more instruction sets for operating the device and the learning the second digital picture correlated with the additional one or more instruction sets for operating the device may include updating a connection between the first node and the second node. In further embodiments, the first digital picture correlated with the one or more instruction sets for operating the device is stored into a first node of a neural network and the second digital picture correlated with the additional one or more instruction sets for operating the device is stored into a second node of the neural network. The first node and the second node may be connected by a connection. The first node may be part of a first layer of the neural network and the second node may be part of a second layer of the neural network. In further embodiments, the first digital picture correlated with the one or more instruction sets for operating the device is stored into a first node of a graph and the second digital picture correlated with the additional one or more instruction sets for operating the device is stored into a second node of the graph. The first node and the second node may be connected by a connection. In further embodiments, the first digital picture correlated with the one or more instruction sets for operating the device is stored into a first node of a sequence and the second digital picture correlated with the additional one or more instruction sets for operating the device is stored into a second node of the sequence.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first digital picture from a picture capturing apparatus. The operations may further include receiving one or more instruction sets for operating a device. The operations may further include learning the first digital picture correlated with the one or more instruction sets for operating the device. The operations may further include receiving a new digital picture from the picture capturing apparatus. The operations may further include anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture. The operations may further include causing an execution of the one or more instruction sets for operating the device correlated with the first digital picture, the causing performed in response to the anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture, wherein the device performs

one or more operations defined by the one or more instruction sets for operating the device correlated with the first digital picture, the one or more operations performed in response to the executing.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first digital picture from a picture capturing apparatus by one or more processor circuits. The method may further include (b) receiving one or more instruction sets for operating a device by the one or more processor circuits. The method may further include (c) learning the first digital picture correlated with the one or more instruction sets for operating the device, the learning of (c) performed by the one or more processor circuits. The method may further include (d) receiving a new digital picture from the picture capturing apparatus by the one or more processor circuits. The method may further include (e) anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture, the anticipating of (e) performed by the one or more processor circuits. The method may further include (f) executing the one or more instruction sets for operating the device correlated with the first digital picture, the executing of (f) performed in response to the anticipating of (e). The method may further include (g) performing, by the device, one or more operations defined by the one or more instruction sets for operating the device correlated with the first digital picture, the one or more operations performed in response to the executing of (f).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems as applicable as well as the following embodiments.

In certain embodiments, the device includes one or more devices. In further embodiments, the device includes a smartphone, a fixture, a control device, a computing enabled device, or a computer. In further embodiments, the picture capturing apparatus includes one or more picture capturing apparatuses. In further embodiments, the picture capturing apparatus includes a motion picture camera or a still picture camera. In further embodiments, the picture capturing apparatus resides on a remote device, the remote device coupled to the one or more processor circuits via a network.

In some embodiments, the one or more instruction sets for operating the device include one or more instruction sets that temporally correspond to the first digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed at a time of the capturing the first digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed prior to the capturing the first digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed within a threshold period of time prior to the capturing the first digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed subsequent to the capturing the first digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed within a threshold period of time subsequent to the capturing the first digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed within a threshold period of time prior to the capturing the first digital picture or a threshold period of time subsequent to the capturing the first digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed from a start of

capturing a preceding digital picture to a start of capturing the first digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed from a start of capturing the first digital picture to a start of capturing a subsequent digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed from a completion of capturing a preceding digital picture to a completion of capturing the first digital picture. The one or more instruction sets that temporally correspond to the first digital picture may include one or more instruction sets executed from a completion of capturing the first digital picture to a completion of capturing a subsequent digital picture.

In certain embodiments, the one or more instruction sets for operating the device are executed by a processor circuit. In further embodiments, the one or more instruction sets for operating the device are part of an application for operating the device. In further embodiments, the one or more instruction sets for operating the device include one or more inputs into or one or more outputs from a processor circuit. In further embodiments, the one or more instruction sets for operating the device include values or states of one or more registers or elements of a processor circuit. In further embodiments, an instruction set includes at least one of: a command, a keyword, a symbol, an instruction, an operator, a variable, a value, an object, a data structure, a function, a parameter, a state, a signal, an input, an output, a character, a digit, or a reference thereto. In further embodiments, the one or more instruction sets include a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, a structured query language (SQL) code, or a machine code. In further embodiments, the one or more instruction sets include one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the one or more instruction sets for operating the device include one or more inputs into a logic circuit. In further embodiments, the one or more instruction sets for operating the device include one or more outputs from a logic circuit. In further embodiments, the one or more instruction sets for operating the device include one or more instruction sets for operating an application or an object of the application.

In some embodiments, the receiving the one or more instruction sets for operating the device includes obtaining the one or more instruction sets. In further embodiments, the receiving the one or more instruction sets for operating the device includes receiving the one or more instruction sets as they are executed. In further embodiments, the receiving the one or more instruction sets for operating the device includes receiving the one or more instruction sets for operating the device from a register or an element of a processor circuit. In further embodiments, the receiving the one or more instruction sets for operating the device includes receiving the one or more instruction sets for operating the device from an element that is part of, operating on, or coupled to a processor circuit. In further embodiments, the receiving the one or more instruction sets for operating the device includes receiving the one or more instruction sets for operating the device from at least one of: a memory unit, the device, a virtual machine, a runtime engine, a hard drive, a storage device, a peripheral device, a network connected device, or a user. In further embodiments, the receiving the one or more instruction sets for operating the device includes receiving the one or more instruction sets from a plurality of processor circuits, applications, memory units, devices, virtual machines, runtime engines, hard drives, storage devices, peripheral devices, network connected devices, or users.

In certain embodiments, the receiving the one or more instruction sets for operating the device includes receiving the one or more instruction sets for operating the device from a logic circuit. The logic circuit may include a microcontroller. The receiving the one or more instruction sets for operating the device from the logic circuit may include receiving the one or more instruction sets for operating the device from an element of the logic circuit. The receiving the one or more instruction sets for operating the device from the logic circuit may include receiving one or more inputs into the logic circuit. The receiving the one or more instruction sets for operating the device from the logic circuit may include receiving one or more outputs from the logic circuit.

In some embodiments, the receiving the one or more instruction sets for operating the device includes receiving the one or more instruction sets for operating the device from an application for operating the device. In further embodiments, the receiving the one or more instruction sets for operating the device includes receiving the one or more instruction sets for operating the device from an application, the application including instruction sets for operating the device. In further embodiments, the receiving the one or more instruction sets for operating the device includes receiving the one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the receiving the one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, a structured query language (SQL) code, or a machine code. In further embodiments, the receiving the one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of an element that is part of, operating on, or coupled to a processor circuit. In further embodiments, the receiving the one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a register of a processor circuit, a memory unit, a storage, or a repository where the one or more instruction sets for operating the device are stored. In further embodiments, the receiving the one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a processor circuit, the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a processing element. In further embodiments, the receiving the one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a processor circuit or tracing, profiling, or instrumentation of a component of the processor circuit. In further embodiments, the receiving the one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of an application or an object of the application. In further embodiments, the receiving the one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the receiving the one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of one or more of code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the receiving the one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a user input. In further embodiments, the receiving the one or more instruction sets for operating the device includes at least one of: a manual, an automatic, a dynamic, or a just in time (JIT) tracing, profiling, or instrumentation. In further embodiments, the receiving the one or more instruction sets for operating the device includes utilizing at least one of: a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, a logging tool, or an independent tool for

obtaining instruction sets. In further embodiments, the receiving the one or more instruction sets for operating the device includes utilizing an assembly language. In further embodiments, the receiving the one or more instruction sets for operating the device includes utilizing a branch or a jump. In further embodiments, the receiving the one or more instruction sets for operating the device includes a branch tracing or a simulation tracing. In further embodiments, the receiving the one or more instruction sets for operating the device includes receiving the one or more instruction sets for operating the device by an interface. The interface may include an acquisition interface.

In certain embodiments, the first digital picture correlated with the one or more instruction sets for operating the device includes a unit of knowledge of how the device operated in a visual surrounding. In further embodiments, the first digital picture correlated with the one or more instruction sets for operating the device is included in a neuron, a node, a vertex, or an element of a data structure. The data structure may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, or a knowledge structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the first digital picture correlated with the one or more instruction sets for operating the device is structured into a knowledge cell. In further embodiments, the knowledge cell includes a unit of knowledge of how the device operated in a visual surrounding. In further embodiments, the knowledge cell is included in a neuron, a node, a vertex, or an element of a data structure. The data structure may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, or a knowledge structure. Some of the neurons, nodes, vertices, or elements may be interconnected.

In certain embodiments, the learning the first digital picture correlated with the one or more instruction sets for operating the device includes correlating the first digital picture with the one or more instruction sets for operating the device. The correlating the first digital picture with the one or more instruction sets for operating the device may include generating a knowledge cell, the knowledge cell comprising the first digital picture correlated with the one or more instruction sets for operating the device. The correlating the first digital picture with the one or more instruction sets for operating the device may include structuring a unit of knowledge of how the device operated in a visual surrounding. In further embodiments, the learning the first digital picture correlated with the one or more instruction sets for operating the device includes learning a user's knowledge, style, or methodology of operating the device in a visual surrounding. In further embodiments, the learning the first digital picture correlated with the one or more instruction sets for operating the device includes spontaneous learning the first digital picture correlated with the one or more instruction sets for operating the device.

In some embodiments, the learning the first digital picture correlated with the one or more instruction sets for operating the device includes storing, into a memory unit, the first digital picture correlated with the one or more instruction sets for operating the device, the first digital picture correlated with the one or more instruction sets for operating the device being part of a stored plurality of digital pictures correlated with one or more instruction sets for operating the device. In further embodiments, the memory unit includes one or more memory units. In further embodiments, the memory unit resides on a remote computing device, the remote computing device coupled to the one or more processor circuits via a network. The remote computing device may include a server. In further embodiments, the plurality of digital pictures correlated with one or more instruction sets for operating the device include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, the plurality of digital pictures

correlated with one or more instruction sets for operating the device are organized into a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, each of the plurality of digital pictures correlated with one or more instruction sets for operating the device is included in a neuron, a node, a vertex, or an element of a data structure. The data structure may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, or a knowledge structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the plurality of digital pictures correlated with one or more instruction sets for operating the device include a user's knowledge, style, or methodology of operating the device in visual surroundings. In further embodiments, the plurality of digital pictures correlated with one or more instruction sets for operating the device are stored on a remote computing device. In further embodiments, the plurality of digital pictures correlated with one or more instruction sets for operating the device include an artificial intelligence system for knowledge structuring, storing, or representation. The artificial intelligence system for knowledge structuring, storing, or representation may include at least one of: a deep learning system, a supervised learning system, an unsupervised learning system, a neural network, a search-based system, an optimization-based system, a logic-based system, a fuzzy logic-based system, a tree-based system, a graph-based system, a hierarchical system, a symbolic system, a sub-symbolic system, an evolutionary system, a genetic system, a multi-agent system, a deterministic system, a probabilistic system, or a statistical system.

In some embodiments, the anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture includes comparing at least one portion of the new digital picture with at least one portion of the first digital picture. The at least one portion of the new digital picture may include at least one region, at least one feature, or at least one pixel of the new digital picture. The at least one portion of the first digital picture may include at least one region, at least one feature, or at least one pixel of the first digital picture. The comparing the at least one portion of the new digital picture with the at least one portion of the first digital picture may include comparing at least one region of the new digital picture with at least one region of the first digital picture. The comparing the at least one portion of the new digital picture with the at least one portion of the first digital picture may include comparing at least one feature of the new digital picture with at least one feature of the first digital picture. The comparing the at least one portion of the new digital picture with the at least one portion of the first digital picture may include comparing at least one pixel of the new digital picture with at least one pixel of the first digital picture. The comparing the at least one portion of the new digital picture with the at least one portion of the first digital picture may include at least one of: performing a color adjustment, performing a size adjustment, performing a content manipulation, utilizing a transparency, or utilizing a mask on the new or the first digital picture. The comparing the at least one portion of the new digital picture with the at least one portion of the first digital picture may include recognizing at least one person or object in the new digital picture and at least one person or object in the first digital picture, and comparing the at least one person or object from the new digital picture with the at least one person or object from the first digital picture.

In certain embodiments, the anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture includes determining that there is at least a partial match between the new digital picture and the first digital

picture. In further embodiments, the determining that there is at least a partial match between the new digital picture and the first digital picture includes determining that there is at least a partial match between one or more portions of the new digital picture and one or more portions of the first digital picture. In further embodiments, the determining that there is at least a partial match between the new digital picture and the first digital picture includes determining that a similarity between at least one portion of the new digital picture and at least one portion of the first digital picture exceeds a similarity threshold. In further embodiments, the determining that there is at least a partial match between the new digital picture and the first digital picture includes determining a substantial similarity between at least one portion of the new digital picture and at least one portion of the first digital picture. The at least one portion of the new digital picture may include at least one region, at least one feature, or at least one pixel of the new digital picture. The at least one portion of the first digital picture may include at least one region, at least one feature, or at least one pixel of the first digital picture. The substantial similarity may be achieved when a similarity between the at least one portion of the new digital picture and the at least one portion of the first digital picture exceeds a similarity threshold. The substantial similarity may be achieved when a number or a percentage of matching or partially matching regions from the new digital picture and from the first digital picture exceeds a threshold number or threshold percentage. The substantial similarity may be achieved when a number or a percentage of matching or partially matching features from the new digital picture and from the first digital picture exceeds a threshold number or threshold percentage. The substantial similarity may be achieved when a number or a percentage of matching or partially matching pixels from the new digital picture and from the first digital picture exceeds a threshold number or threshold percentage. The substantial similarity may be achieved when one or more same or similar objects are recognized in the new digital picture and the first digital picture. In further embodiments, the determining that there is at least a partial match between the new digital picture and the first digital picture includes determining that a number or a percentage of matching regions from the new digital picture and from the first digital picture exceeds a threshold number or threshold percentage. The matching regions from the new digital picture and from the first digital picture may be determined factoring in at least one of: a location of a region, an importance of a region, a threshold for a similarity in a region, or a threshold for a difference in a region. In further embodiments, the determining that there is at least a partial match between the new digital picture and the first digital picture includes determining that a number or a percentage of matching features from the new digital picture and from the first digital picture exceeds a threshold number or threshold percentage. The matching features from the new digital picture and from the first digital picture may be determined factoring in at least one of: a type of a feature, an importance of a feature, a location of a feature, a threshold for a similarity in a feature, or a threshold for a difference in a feature. In further embodiments, the determining that there is at least a partial match between the new digital picture and the first digital picture includes determining that a number or a percentage of matching pixels from the new digital picture and from the first digital picture exceeds a threshold number or threshold percentage. The matching pixels from the new digital picture and from the first digital picture may be determined factoring in at least one of: a location of a pixel, a threshold for a similarity in a pixel, or a threshold for a difference in a pixel. In further embodiments, the determining that there is at least a partial match between the new digital picture and the first digital picture includes recognizing a same person or object in the new and the first digital pictures.

In some embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes executing the one or more instruction sets for operating the device correlated

with the first digital picture instead of or prior to an instruction set that would have been executed next. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes modifying one or more instruction sets. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes modifying a register or an element of a processor circuit. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes inserting the one or more instruction sets for operating the device correlated with the first digital picture into a register or an element of a processor circuit. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes redirecting a processor circuit to the one or more instruction sets for operating the device correlated with the first digital picture. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes redirecting a processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the one or more instruction sets for operating the device correlated with the first digital picture. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes transmitting, to a processor circuit for execution, the one or more instruction sets for operating the device correlated with the first digital picture. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes issuing an interrupt to a processor circuit and executing the one or more instruction sets for operating the device correlated with the first digital picture following the interrupt. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes modifying an element that is part of, operating on, or coupled to a processor circuit.

In certain embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes executing, by a logic circuit, the one or more instruction sets for operating the device correlated with the first digital picture. The logic circuit may include a microcontroller. The executing, by the logic circuit, the one or more instruction sets for operating the device correlated with the first digital picture may include modifying an element of the logic circuit. The executing, by the logic circuit, the one or more instruction sets for operating the device correlated with the first digital picture may include inserting the one or more instruction sets for operating the device correlated with the first digital picture into an element of the logic circuit. The executing, by the logic circuit, the one or more instruction sets for operating the device correlated with the first digital picture may include redirecting the logic circuit to the one or more instruction sets for operating the device correlated with the first digital picture. The executing, by the logic circuit, the one or more instruction sets for operating the device correlated with the first digital picture may include replacing inputs into the logic circuit with the one or more instruction sets for operating the device correlated with the first digital picture. The executing, by the logic circuit, the one or more instruction sets for operating the device correlated with the first digital picture may include replacing outputs from the logic circuit with the one or more instruction sets for operating the device correlated with the first digital picture.

In some embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes executing, by an application for operating the device, the one or more instruction sets for operating the device correlated with the first digital picture. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes modifying an application, the application including instruction sets for operating the device. In further embodiments, the executing

the one or more instruction sets for operating the device correlated with the first digital picture includes redirecting an application to the one or more instruction sets for operating the device correlated with the first digital picture. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes redirecting an application to one or more alternate instruction sets, the alternate instruction sets comprising the one or more instruction sets for operating the device correlated with the first digital picture. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes modifying one or more instruction sets of an application. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes modifying at least one of: a memory unit, a register of a processor circuit, a storage, or a repository where instruction sets are stored or used. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes modifying one or more instruction sets for operating an application or an object of the application. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes modifying at least one of: an element of a processor circuit, an element of the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes modifying one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of an application. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes utilizing an assembly language. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes utilizing at least one of: a dynamic expression

creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes adding or inserting additional code into a code of an application. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes at least one of: modifying, removing, rewriting, or overwriting a code of an application. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes at least one of: branching, redirecting, extending, or hot swapping a code of an application. The branching or redirecting the code may include inserting at least one of: a branch, a jump, or a means for redirecting an execution. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes implementing a user's knowledge, style, or methodology of operating the device in a visual surrounding. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first digital picture includes executing the one or more instruction sets for operating the device correlated with the first digital picture via an interface. The interface may include a modification interface.

In certain embodiments, the one or more operations defined by the one or more instruction sets for operating the device correlated with the first digital picture include at least one of: an operation with or by a smartphone, an operation with or by a fixture, an operation with or by a control device , or an operation with or by a computer or computing enabled device. In further embodiments, the performing the one or more operations defined by the one or more instruction sets for operating the device correlated with the first digital picture includes implementing a user's knowledge, style, or methodology of operating the device in a visual surrounding.

In some embodiments, the instruction sets for operating the device are part of an application for operating the device.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving at least one extra information. In further embodiments, the at least one extra information include one or more of: a time information, a location information, a computed information, an observed information, a sensory information, or a contextual information. In further embodiments, the at least one extra information include one or more of: an information on a digital picture, an information on an object in the digital picture, an information on the device's visual surrounding, an information on an instruction set, an information on an application, an information on an object of the application, an information on a processor circuit, an information on the device, or an information on an user. In further embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: learning the first digital picture correlated with the at least one extra information. The learning the first digital picture correlated with at least one extra information may include correlating the first digital picture with the at least one extra information. The learning the first digital picture correlated with at least one extra information may include storing the first digital picture correlated with the at least one extra information into a memory unit. In further embodiments, the anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture includes anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between an extra information correlated with the new digital picture and an extra information correlated with the first digital picture. The anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between an extra

information correlated with the new digital picture and an extra information correlated with the first digital picture may include comparing an extra information correlated with the new digital picture and an extra information correlated with the first digital picture. The anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between an extra information correlated with the new digital picture and an extra information correlated with the first digital picture may include determining that a similarity between an extra information correlated with the new digital picture and an extra information correlated with the first digital picture exceeds a similarity threshold.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: presenting, via a user interface, a user with an option to execute the one or more instruction sets for operating the device correlated with the first digital picture.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving, via a user interface, a user's selection to execute the one or more instruction sets for operating the device correlated with the first digital picture.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: rating the executed one or more instruction sets for operating the device correlated with the first digital picture. The rating the executed one or more instruction sets for operating the device correlated with the first digital picture may include displaying, on a display, the executed one or more instruction sets for operating the device correlated with the first digital picture along with one or more rating values as options to be selected by a user. The rating the executed one or more instruction sets for operating the device correlated with the first digital picture may include rating the executed one or more instruction sets for operating the device correlated with the first digital picture without a user input. The rating the executed one or more instruction sets for operating the device correlated with the first digital picture may include associating one or more rating values with the executed one or more instruction sets for operating the device correlated with the first digital picture and storing the one or more rating values into a memory unit.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: presenting, via a user interface, a user with an option to cancel the execution of the executed one or more instruction sets for operating the device correlated with the first digital picture. In further embodiments, the canceling the execution of the executed one or more instruction sets for operating the device correlated with the first digital picture includes restoring a processor circuit or the device to a prior state. The restoring the processor circuit or the device to a prior state may include saving the state of the processor circuit or the device prior to executing the one or more instruction sets for operating the device correlated with the first digital picture.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving, via an input device, a user's operating directions, the user's operating directions for instructing a processor circuit on how to operate the device.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving a second digital picture from the picture capturing apparatus; receiving additional one or more instruction sets for operating the device; and learning the second digital picture correlated with the additional one or more instruction sets for operating the device. In further embodiments, the second digital picture includes a second stream of digital pictures. In further embodiments, the learning the first digital picture correlated with the one

or more instruction sets for operating the device and the learning the second digital picture correlated with the additional one or more instruction sets for operating the device include creating a connection between the first digital picture correlated with the one or more instruction sets for operating the device and the second digital picture correlated with the additional one or more instruction sets for operating the device. The connection may include or is associated with at least one of: an occurrence count, a weight, a parameter, or a data. In further embodiments, the learning the first digital picture correlated with the one or more instruction sets for operating the device and the learning the second digital picture correlated with the additional one or more instruction sets for operating the device include updating a connection between the first digital picture correlated with the one or more instruction sets for operating the device and the second digital picture correlated with the additional one or more instruction sets for operating the device. The updating the connection between the first digital picture correlated with the one or more instruction sets for operating the device and the second digital picture correlated with the additional one or more instruction sets for operating the device may include updating at least one of: an occurrence count, a weight, a parameter, or a data included in or associated with the connection. In further embodiments, the learning the first digital picture correlated with the one or more instruction sets for operating the device includes storing the first digital picture correlated with the one or more instruction sets for operating the device into a first node of a data structure, and wherein the learning the second digital picture correlated with the additional one or more instruction sets for operating the device may include storing the second digital picture correlated with the additional one or more instruction sets for operating the device into a second node of the data structure. The data structure may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, or a knowledge structure. The learning the first digital picture correlated with the one or more instruction sets for operating the device and the learning the second digital picture correlated with the additional one or more instruction sets for operating the device may include creating a connection between the first node and the second node. The learning the first digital picture correlated with the one or more instruction sets for operating the device and the learning the second digital picture correlated with the additional one or more instruction sets for operating the device may include updating a connection between the first node and the second node. In further embodiments, the first digital picture correlated with the one or more instruction sets for operating the device is stored into a first node of a neural network and the second digital picture correlated with the additional one or more instruction sets for operating the device is stored into a second node of the neural network. The first node and the second node may be connected by a connection. The first node may be part of a first layer of the neural network and the second node may be part of a second layer of the neural network. In further embodiments, the first digital picture correlated with the one or more instruction sets for operating the device is stored into a first node of a graph and the second digital picture correlated with the additional one or more instruction sets for operating the device is stored into a second node of the graph. The first node and the second node may be connected by a connection. In further embodiments, the first digital picture correlated with the one or more instruction sets for operating the device is stored into a first node of a sequence and the second digital picture correlated with the additional one or more instruction sets for operating the device is stored into a second node of the sequence.

In some aspects, the disclosure relates to a system for learning a visual surrounding for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises a processor circuit configured to execute instruction sets for operating a

device. The system may further include a memory unit configured to store data. The system may further include a picture capturing apparatus configured to capture digital pictures. The system may further include an artificial intelligence unit. In some embodiments, the artificial intelligence unit may be configured to: receive a first digital picture from the picture capturing apparatus. The artificial intelligence unit may be further configured to: receive one or more instruction sets for operating the device from the processor circuit. The artificial intelligence unit may be further configured to: learn the first digital picture correlated with the one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first digital picture from a picture capturing apparatus. The operations may further include: receiving one or more instruction sets for operating a device. The operations may further include: learning the first digital picture correlated with the one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first digital picture from a picture capturing apparatus by one or more processor circuits. The method may further include: (b) receiving one or more instruction sets for operating a device by the one or more processor circuits. The method may further include: (c) learning the first digital picture correlated with the one or more instruction sets for operating the device, the learning of (c) performed by the one or more processor circuits.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems as applicable.

In some aspects, the disclosure relates to a system for using a visual surrounding for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises a processor circuit configured to execute instruction sets for operating a device. The system may further include a memory unit configured to store data. The system may further include a picture capturing apparatus configured to capture digital pictures. The system may further include an artificial intelligence unit. In some embodiments, the artificial intelligence unit may be configured to: access the memory unit that stores a plurality of digital pictures correlated with one or more instruction sets for operating the device, the plurality including a first digital picture correlated with one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: receive a new digital picture from the picture capturing apparatus. The artificial intelligence unit may be further configured to: anticipate the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture, the executing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the one or more instruction sets for operating the device correlated with the first digital picture, the one or more operations performed in response to the executing by the processor circuit.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: accessing a memory unit that stores a plurality of digital pictures correlated with one or more instruction sets for operating a device, the plurality including a first digital picture correlated with one or more instruction sets for operating the device. The operations may further include: receiving a new digital picture from a picture capturing apparatus. The operations may further include: anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture. The operations may further include: causing an execution of the one or more instruction sets for operating the device correlated with the first digital picture, the causing performed in response to the anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture, wherein the device performs one or more operations defined by the one or more instruction sets for operating the device correlated with the first digital picture, the one or more operations performed in response to the executing.

In some aspects, the disclosure relates to a method comprising: (a) accessing a memory unit that stores a plurality of digital pictures correlated with one or more instruction sets for operating a device, the plurality including a first digital picture correlated with one or more instruction sets for operating the device, the accessing of (a) performed by the one or more processor circuits. The method may further include: (b) receiving a new digital picture from a picture capturing apparatus by the one or more processor circuits. The method may further include: (c) anticipating the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture, the anticipating of (c) performed by the one or more processor circuits. The method may further include: (d) executing the one or more instruction sets for operating the device correlated with the first digital picture, the executing of (d) performed in response to the anticipating of (c). The method may further include: (e) performing, by the device, one or more operations defined by the one or more instruction sets for operating the device correlated with the first digital picture, the one or more operations performed in response to the executing of (d).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems as applicable.

In some aspects, the disclosure relates to a system for learning and using a visual surrounding for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises a processor circuit configured to execute instruction sets for operating a device. The system may further include a memory unit configured to store data. The system may further include a picture capturing apparatus configured to capture digital pictures. The system may further include an artificial intelligence. In some embodiments, the artificial intelligence unit may be configured to: receive a first stream of digital pictures from the picture capturing apparatus. The artificial intelligence unit may be further configured to: receive one or more instruction sets for operating the device from the processor circuit. The artificial intelligence unit may be further configured to: learn the first stream of digital pictures correlated with the one or more instruction sets

for operating the device. The artificial intelligence unit may be further configured to: receive a new stream of digital pictures from the picture capturing apparatus. The artificial intelligence unit may be further configured to: anticipate the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between the new stream of digital pictures and the first stream of digital pictures. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures, the executing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the one or more instruction sets for operating the device correlated with the first stream of digital pictures, the one or more operations performed in response to the executing by the processor circuit.

In certain embodiments, the first stream of digital pictures includes one or more digital pictures. In further embodiments, the new stream of digital pictures includes one or more digital pictures. In further embodiments, the first and the new streams of digital pictures portray the device's surrounding. In further embodiments, the first and the new streams of digital pictures portray a remote device's surrounding. In further embodiments, the first or the new stream of digital pictures includes a digital motion picture. The digital motion picture may include a MPEG motion picture, an AVI motion picture, a FLV motion picture, a MOV motion picture, a RM motion picture, a SWF motion picture, a WMV motion picture, a DivX motion picture, or a digitally encoded motion picture. In further embodiments, the first stream of digital pictures includes a comparative stream of digital pictures whose at least one portion can be used for comparisons with at least one portion of streams of digital pictures subsequent to the first stream of digital pictures, the streams of digital pictures subsequent to the first stream of digital pictures comprising the new stream of digital pictures. In further embodiments, the first stream of digital pictures includes a comparative stream of digital pictures that can be used for comparisons with the new stream of digital pictures. In further embodiments, the new stream of digital pictures includes an anticipatory stream of digital pictures whose correlated one or more instruction sets can be used for anticipation of one or more instruction sets to be executed by the processor circuit.

In some embodiments, the first stream of digital pictures correlated with the one or more instruction sets for operating the device includes a unit of knowledge of how the device operated in a visual surrounding. In further embodiments, the first stream of digital pictures correlated with the one or more instruction sets for operating the device is included in a neuron, a node, a vertex, or an element of a data structure. In further embodiments, the data structure includes a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, or a knowledge structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the first stream of digital pictures correlated with the one or more instruction sets for operating the device is structured into a knowledge cell. In further embodiments, the knowledge cell includes a unit of knowledge of how the device operated in a visual surrounding. In further embodiments, the knowledge cell is included in a neuron, a node, a vertex, or an element of a data structure. The data structure may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, or a knowledge structure. Some of the neurons, nodes, vertices, or elements may be interconnected.

In certain embodiments, the learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device includes correlating the first stream of digital pictures with the one or more instruction sets for operating the device. The correlating the first stream of digital pictures with the one or more

instruction sets for operating the device may include generating a knowledge cell, the knowledge cell comprising the first stream of digital pictures correlated with the one or more instruction sets for operating the device. The correlating the first stream of digital pictures with the one or more instruction sets for operating the device may include structuring a unit of knowledge of how the device operated in a visual surrounding. In further embodiments, the learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device includes learning a user's knowledge, style, or methodology of operating the device in a visual surrounding. In further embodiments, the learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device includes spontaneous learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device.

In some embodiments, the learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device includes storing, into the memory unit, the first stream of digital pictures correlated with the one or more instruction sets for operating the device, the first stream of digital pictures correlated with the one or more instruction sets for operating the device being part of a stored plurality of streams of digital pictures correlated with one or more instruction sets for operating the device. In further embodiments, the plurality of streams of digital pictures correlated with one or more instruction sets for operating the device include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, the plurality of streams of digital pictures correlated with one or more instruction sets for operating the device are organized into a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, each of the plurality of streams of digital pictures correlated with one or more instruction sets for operating the device is included in a neuron, a node, a vertex, or an element of a data structure. The data structure may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, or a knowledge structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the plurality of streams of digital pictures correlated with one or more instruction sets for operating the device include a user's knowledge, style, or methodology of operating the device in visual surroundings. In further embodiments, the plurality of streams of digital pictures correlated with one or more instruction sets for operating the device are stored on a remote computing device. In further embodiments, the plurality of streams of digital pictures correlated with one or more instruction sets for operating the device include an artificial intelligence system for knowledge structuring, storing, or representation. The artificial intelligence system for knowledge structuring, storing, or representation may include at least one of: a deep learning system, a supervised learning system, an unsupervised learning system, a neural network, a search-based system, an optimization-based system, a logic-based system, a fuzzy logic-based system, a tree-based system, a graph-based system, a hierarchical system, a symbolic system, a sub-symbolic system, an evolutionary system, a genetic system, a multi-agent system, a deterministic system, a probabilistic system, or a statistical system.

In some embodiments, the anticipating the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes comparing at least one portion of the new stream of digital pictures with at least one portion of the first stream of digital pictures. The at least one portion of the new stream of digital pictures may include at least one digital picture, at least one region, at least one feature, or at least one pixel of the

new stream of digital pictures. The at least one portion of the first stream of digital pictures may include at least one digital picture, at least one region, at least one feature, or at least one pixel of the first stream of digital pictures. The comparing the at least one portion of the new stream of digital pictures with the at least one portion of the first stream of digital pictures may include comparing at least one digital picture of the new stream of digital pictures with at least one digital picture of the first stream of digital pictures. The at least one portion of the new stream of digital pictures with the at least one portion of the first stream of digital pictures may include comparing at least one region of at least one digital picture of the new stream of digital pictures with at least one region of at least one digital picture of the first stream of digital pictures. The comparing the at least one portion of the new stream of digital pictures with the at least one portion of the first stream of digital pictures may include comparing at least one feature of at least one digital picture of the new stream of digital pictures with at least one feature of at least one digital picture of the first stream of digital pictures. The comparing the at least one portion of the new stream of digital pictures with the at least one portion of the first stream of digital pictures may include comparing at least one pixel of at least one digital picture of the new stream of digital pictures with at least one pixel of at least one digital picture of the first stream of digital pictures. The comparing the at least one portion of the new stream of digital pictures with the at least one portion of the first stream of digital pictures may include at least one of: performing a color adjustment, performing a size adjustment, performing a content manipulation, performing temporal alignment, performing dynamic time warping, utilizing a transparency, or utilizing a mask on the new or the first stream of digital pictures. The comparing the at least one portion of the new stream of digital pictures with the at least one portion of the first stream of digital pictures may include recognizing at least one person or object in the new stream of digital pictures and at least one person or object in the first stream of digital pictures, and comparing the at least one person or object from the new stream of digital pictures with the at least one person or object from the first stream of digital pictures.

In certain embodiments, the anticipating the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes determining that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures. In further embodiments, the determining that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes determining that there is at least a partial match between one or more portions of the new stream of digital pictures and one or more portions of the first stream of digital pictures. In further embodiments, the determining that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes determining that a similarity between at least one portion of the new stream of digital pictures and at least one portion of the first stream of digital pictures exceeds a similarity threshold. In further embodiments, the determining that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes determining a substantial similarity between at least one portion of the new stream of digital pictures and at least one portion of the first stream of digital pictures. The at least one portion of the new stream of digital pictures may include at least one digital picture, at least one region, at least one feature, or at least one pixel of the new stream of digital pictures. The at least one portion of the first stream of digital pictures may include at least one digital picture, at least one region, at least one feature, or at least one pixel of the first stream of digital pictures. The substantial similarity may be achieved when a similarity between the at least one portion of the new stream of digital

pictures and the at least one portion of the first stream of digital pictures exceeds a similarity threshold. The substantial similarity may be achieved when a number or a percentage of matching or partially matching digital pictures from the new stream of digital pictures and from the first stream of digital pictures exceeds a threshold number or threshold percentage. The substantial similarity may be achieved when a number or a percentage of matching or partially matching regions of at least one digital picture from the new stream of digital pictures and from the first stream of digital pictures exceeds a threshold number or threshold percentage. The substantial similarity may be achieved when a number or a percentage of matching or partially matching features of at least one digital picture from the new stream of digital pictures and from the first stream of digital pictures exceeds a threshold number or threshold percentage. The substantial similarity may be achieved when a number or a percentage of matching or partially matching pixels of at least one digital picture from the new stream of digital pictures and from the first stream of digital pictures exceeds a threshold number or threshold percentage. The substantial similarity may be achieved when one or more same or similar objects are recognized in the new stream of digital pictures and the first stream of digital pictures. In further embodiments, the determining that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes determining that a number or a percentage of matching digital pictures from the new stream of digital pictures and from the first stream of digital pictures exceeds a threshold number or threshold percentage. The matching digital pictures from the new stream of digital pictures and from the first stream of digital pictures may be determined factoring in at least one of: an order of a digital picture in a stream of digital pictures, an importance of a digital picture, a threshold for a similarity in a digital picture, or a threshold for a difference in a digital picture. In further embodiments, the determining that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes determining that a number or a percentage of matching regions from at least one digital picture of the new stream of digital pictures and from at least one digital picture of the first stream of digital pictures exceeds a threshold number or threshold percentage. The matching regions from at least one digital picture of the new stream of digital pictures and from at least one digital picture of the first stream of digital pictures may be determined factoring in at least one of: a location of a region, an importance of a region, a threshold for a similarity in a region, or a threshold for a difference in a region. In further embodiments, the determining that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes determining that a number or a percentage of matching features from at least one digital picture of the new stream of digital pictures and from at least one digital picture of the first stream of digital pictures exceeds a threshold number or threshold percentage. The matching features from at least one digital picture of the new stream of digital pictures and from at least one digital picture of the first stream of digital pictures may be determined factoring in at least one of: a type of a feature, an importance of a feature, a location of a feature, a threshold for a similarity in a feature, or a threshold for a difference in a feature. In further embodiments, the determining that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes determining that a number or a percentage of matching pixels from at least one digital picture of the new stream of digital pictures and from at least one digital picture of the first stream of digital pictures exceeds a threshold number or threshold percentage. The matching pixels from at least one digital picture of the new stream of digital pictures and from at least one digital picture of the first stream of digital pictures may be determined factoring in at least one of: a location of a pixel, a threshold for a similarity in a pixel, or a threshold for a difference in a pixel. In further embodiments, the determining that there is at

least a partial match between the new stream of digital pictures and the first stream of digital pictures includes recognizing a same person or object in the new and the first streams of digital pictures.

In certain embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures instead of or prior to an instruction set that would have been executed next. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying one or more instruction sets of the processor circuit. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes inserting the one or more instruction sets for operating the device correlated with the first stream of digital pictures into a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes redirecting the processor circuit to the one or more instruction sets for operating the device correlated with the first stream of digital pictures. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes redirecting the processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the one or more instruction sets for operating the device correlated with the first stream of digital pictures. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes transmitting, to the processor circuit for execution, the one or more instruction sets for operating the device correlated with the first stream of digital pictures. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes issuing an interrupt to the processor circuit and executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures following the interrupt. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying an element that is part of, operating on, or coupled to the processor circuit.

In some embodiments, the processor circuit includes a logic circuit, and wherein the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes causing the logic circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures. The logic circuit may include a microcontroller. The causing the logic circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures may include modifying an element of the logic circuit. The causing the logic circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures may include inserting the one or more instruction sets for operating the device correlated with the first stream of digital pictures into an element of the logic circuit. The causing the logic circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures may include redirecting the logic circuit to the one or more instruction sets for operating the device correlated with the first stream of digital pictures. The causing

the logic circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures may include replacing inputs into the logic circuit with the one or more instruction sets for operating the device correlated with the first stream of digital pictures. The causing the logic circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures may include replacing outputs from the logic circuit with the one or more instruction sets for operating the device correlated with the first stream of digital pictures.

In certain embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes causing an application for operating the device to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures, the application running on the processor circuit.

In some embodiments, the system further comprises: an application including instruction sets for operating the device, the application running on the processor circuit, wherein the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying the application.

In certain embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes redirecting an application to the one or more instruction sets for operating the device correlated with the first stream of digital pictures, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes redirecting an application to one or more alternate instruction sets, the application running on the processor circuit, the alternate instruction sets comprising the one or more instruction sets for operating the device correlated with the first stream of digital pictures. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying one or more instruction sets of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying at least one of: the memory unit, a register of the processor circuit, a storage, or a repository where instruction sets are stored or used. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying one or more instruction sets for operating an application or an object of the application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying at least one of: an element of the processor circuit, an element of the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime.

In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes utilizing an assembly language. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes adding or inserting additional code into a code of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes at least one of: modifying, removing, rewriting, or overwriting a code of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes at least one of: branching, redirecting, extending, or hot swapping a code of an application, the application running on the processor circuit. The branching or redirecting the code may include inserting at least one of: a branch, a jump, or a means for redirecting an execution.

In some embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes implementing a user's knowledge, style, or methodology of operating the device in a visual surrounding.

In certain embodiments, the system further comprises: an interface configured to cause execution of instruction sets, wherein the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures is caused by the interface. The interface may include a modification interface.

In some embodiments, the artificial intelligence unit is further configured to: receive at least one extra information. In further embodiments, the at least one extra information include one or more of: an information on a stream of digital pictures, an information on an object in the stream of digital pictures, an information on the device's visual surrounding, an information on an instruction set, an information on an application, an information on an object of the application, an information on the processor circuit, an information on the device, or an information on an user. In further embodiments, the artificial intelligence unit is further configured to: learn the first stream of digital pictures correlated with the at least one extra information. The learning the first stream of digital pictures correlated with at least one extra information may include correlating the first stream of digital pictures with the at least one extra information. The learning the first stream of digital pictures correlated with at least one extra information may include storing the first stream of digital pictures correlated with the at least one extra information into the memory unit. In further embodiments, the anticipating the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes anticipating the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between an extra information correlated with the new stream of digital pictures and an extra information correlated with the first stream of digital pictures. The anticipating the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between an extra information correlated with the new stream of digital pictures and an extra information correlated with the first stream of digital pictures may include comparing an extra information correlated with the new stream of digital pictures and an extra information correlated with the first stream of digital pictures. The anticipating the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between an extra information correlated with the new stream of digital pictures and an extra information correlated with the first stream of digital pictures may include determining that a similarity between an extra information correlated with the new stream of digital pictures and an extra information correlated with the first stream of digital pictures exceeds a similarity threshold.

In certain embodiments, the artificial intelligence unit is further configured to: receive a second stream of digital pictures from the picture capturing apparatus; receive additional one or more instruction sets for operating the device from the processor circuit; and learn the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device. In further embodiments, the learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device and the learning the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device include creating a connection between the first stream of digital pictures correlated with the one or more instruction sets for operating the device and the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device. The connection includes or is associated with at least one of: an occurrence count, a weight, a parameter, or a data. In further embodiments, the learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device and the learning the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device include updating a connection between the first stream of digital pictures correlated with the one or more instruction sets for operating the device and the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device. The updating the connection between the first stream of digital pictures correlated with the one or more instruction

sets for operating the device and the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device may include updating at least one of: an occurrence count, a weight, a parameter, or a data included in or associated with the connection. In further embodiments, the learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device includes storing the first stream of digital pictures correlated with the one or more instruction sets for operating the device into a first node of a data structure, and wherein the learning the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device includes storing the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device into a second node of the data structure. The data structure may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, or a knowledge structure. The learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device and the learning the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device may include creating a connection between the first node and the second node. The learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device and the learning the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device may include updating a connection between the first node and the second node. In further embodiments, the first stream of digital pictures correlated with the one or more instruction sets for operating the device is stored into a first node of a neural network and the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device is stored into a second node of the neural network. The first node and the second node may be connected by a connection. The first node may be part of a first layer of the neural network and the second node may be part of a second layer of the neural network. In further embodiments, the first stream of digital pictures correlated with the one or more instruction sets for operating the device is stored into a first node of a graph and the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device is stored into a second node of the graph. The first node and the second node may be connected by a connection. In further embodiments, the first stream of digital pictures correlated with the one or more instruction sets for operating the device is stored into a first node of a sequence and the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device is stored into a second node of the sequence.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first stream of digital pictures from a picture capturing apparatus. The operations may further include: receiving one or more instruction sets for operating a device. The operations may further include: learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device. The operations may further include: receiving a new stream of digital pictures from the picture capturing apparatus. The operations may further include: anticipating the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between the new stream of digital pictures and the first stream of digital pictures. The operations may further include: causing an execution of the one or more instruction sets for operating the device correlated with the first stream of digital pictures, the causing performed in response to the anticipating the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between

the new stream of digital pictures and the first stream of digital pictures, wherein the device performs one or more operations defined by the one or more instruction sets for operating the device correlated with the first stream of digital pictures, the one or more operations performed in response to the executing.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first stream of digital pictures from a picture capturing apparatus by one or more processor circuits. The method may further include: (b) receiving one or more instruction sets for operating a device by the one or more processor circuits. The method may further include: (c) learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device, the learning of (c) performed by the one or more processor circuits. The method may further include: (d) receiving a new stream of digital pictures from the picture capturing apparatus by the one or more processor circuits. The method may further include: (e) anticipating the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between the new stream of digital pictures and the first stream of digital pictures, the anticipating of (e) performed by the one or more processor circuits. The method may further include: (f) executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures, the executing of (f) performed in response to the anticipating of (e). The method may further include: (g) performing, by the device, one or more operations defined by the one or more instruction sets for operating the device correlated with the first stream of digital pictures, the one or more operations performed in response to the executing of (f).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems as applicable as well as the following embodiments.

In some embodiments, the first stream of digital pictures correlated with the one or more instruction sets for operating the device includes a unit of knowledge of how the device operated in a visual surrounding. In further embodiments, the first stream of digital pictures correlated with the one or more instruction sets for operating the device is included in a neuron, a node, a vertex, or an element of a data structure. The data structure may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, or a knowledge structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the first stream of digital pictures correlated with the one or more instruction sets for operating the device is structured into a knowledge cell. In further embodiments, the knowledge cell includes a unit of knowledge of how the device operated in a visual surrounding. In further embodiments, the knowledge cell is included in a neuron, a node, a vertex, or an element of a data structure. The data structure may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, or a knowledge structure. Some of the neurons, nodes, vertices, or elements may be interconnected.

In certain embodiments, the learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device includes correlating the first stream of digital pictures with the one or more instruction sets for operating the device. The correlating the first stream of digital pictures with the one or more instruction sets for operating the device may include generating a knowledge cell, the knowledge cell comprising the first stream of digital pictures correlated with the one or more instruction sets for operating the device. The correlating the first stream of digital pictures with the one or more instruction sets for operating the device may

include structuring a unit of knowledge of how the device operated in a visual surrounding. In further embodiments, the learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device includes learning a user's knowledge, style, or methodology of operating the device in a visual surrounding. In further embodiments, the learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device includes spontaneous learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device.

In some embodiments, the learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device includes storing, into a memory unit, the first stream of digital pictures correlated with the one or more instruction sets for operating the device, the first stream of digital pictures correlated with the one or more instruction sets for operating the device being part of a stored plurality of streams of digital pictures correlated with one or more instruction sets for operating the device. In further embodiments, the plurality of streams of digital pictures correlated with one or more instruction sets for operating the device include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, the plurality of streams of digital pictures correlated with one or more instruction sets for operating the device are organized into a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, each of the plurality of streams of digital pictures correlated with one or more instruction sets for operating the device is included in a neuron, a node, a vertex, or an element of a data structure. The data structure may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, or a knowledge structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the plurality of streams of digital pictures correlated with one or more instruction sets for operating the device include a user's knowledge, style, or methodology of operating the device in visual surroundings. In further embodiments, the plurality of streams of digital pictures correlated with one or more instruction sets for operating the device are stored on a remote computing device. In further embodiments, the plurality of streams of digital pictures correlated with one or more instruction sets for operating the device include an artificial intelligence system for knowledge structuring, storing, or representation. The artificial intelligence system for knowledge structuring, storing, or representation may include at least one of: a deep learning system, a supervised learning system, an unsupervised learning system, a neural network, a search-based system, an optimization-based system, a logic-based system, a fuzzy logic-based system, a tree-based system, a graph-based system, a hierarchical system, a symbolic system, a sub-symbolic system, an evolutionary system, a genetic system, a multi-agent system, a deterministic system, a probabilistic system, or a statistical system.

In further embodiments, the anticipating the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes comparing at least one portion of the new stream of digital pictures with at least one portion of the first stream of digital pictures. The at least one portion of the new stream of digital pictures may include at least one digital picture, at least one region, at least one feature, or at least one pixel of the new stream of digital pictures. The at least one portion of the first stream of digital pictures may include at least one digital picture, at least one region, at least one feature, or at least one pixel of the first stream of digital pictures. The comparing the at least one portion of the new stream of digital pictures with the at least one portion of the first

stream of digital pictures may include comparing at least one digital picture of the new stream of digital pictures with at least one digital picture of the first stream of digital pictures. The comparing the at least one portion of the new stream of digital pictures with the at least one portion of the first stream of digital pictures may include comparing at least one region of at least one digital picture of the new stream of digital pictures with at least one region of at least one digital picture of the first stream of digital pictures. The comparing the at least one portion of the new stream of digital pictures with the at least one portion of the first stream of digital pictures may include comparing at least one feature of at least one digital picture of the new stream of digital pictures with at least one feature of at least one digital picture of the first stream of digital pictures. The comparing the at least one portion of the new stream of digital pictures with the at least one portion of the first stream of digital pictures may include comparing at least one pixel of at least one digital picture of the new stream of digital pictures with at least one pixel of at least one digital picture of the first stream of digital pictures. The comparing the at least one portion of the new stream of digital pictures with the at least one portion of the first stream of digital pictures may include at least one of: performing a color adjustment, performing a size adjustment, performing a content manipulation, performing temporal alignment, performing dynamic time warping, utilizing a transparency, or utilizing a mask on the new or the first stream of digital pictures. The comparing the at least one portion of the new stream of digital pictures with the at least one portion of the first stream of digital pictures may include recognizing at least one person or object in the new stream of digital pictures and at least one person or object in the first stream of digital pictures, and comparing the at least one person or object from the new stream of digital pictures with the at least one person or object from the first stream of digital pictures.

In some embodiments, the anticipating the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes determining that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures. In further embodiments, the determining that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes determining that there is at least a partial match between one or more portions of the new stream of digital pictures and one or more portions of the first stream of digital pictures. In further embodiments, the determining that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes determining that a similarity between at least one portion of the new stream of digital pictures and at least one portion of the first stream of digital pictures exceeds a similarity threshold. In further embodiments, the determining that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes determining a substantial similarity between at least one portion of the new stream of digital pictures and at least one portion of the first stream of digital pictures. The at least one portion of the new stream of digital pictures may include at least one digital picture, at least one region, at least one feature, or at least one pixel of the new stream of digital pictures. The at least one portion of the first stream of digital pictures may include at least one digital picture, at least one region, at least one feature, or at least one pixel of the first stream of digital pictures. The substantial similarity may be achieved when a similarity between the at least one portion of the new stream of digital pictures and the at least one portion of the first stream of digital pictures exceeds a similarity threshold. The substantial similarity may be achieved when a number or a percentage of matching or partially matching digital pictures from the new stream of digital pictures and from the first stream of digital pictures exceeds a threshold

number or threshold percentage. The substantial similarity may be achieved when a number or a percentage of matching or partially matching regions of at least one digital picture from the new stream of digital pictures and from the first stream of digital pictures exceeds a threshold number or threshold percentage. The substantial similarity may be achieved when a number or a percentage of matching or partially matching features of at least one digital picture from the new stream of digital pictures and from the first stream of digital pictures exceeds a threshold number or threshold percentage. The substantial similarity may be achieved when a number or a percentage of matching or partially matching pixels of at least one digital picture from the new stream of digital pictures and from the first stream of digital pictures exceeds a threshold number or threshold percentage. The substantial similarity may be achieved when one or more same or similar objects are recognized in the new stream of digital pictures and the first stream of digital pictures. In further embodiments, the determining that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes determining that a number or a percentage of matching digital pictures from the new stream of digital pictures and from the first stream of digital pictures exceeds a threshold number or threshold percentage. The matching digital pictures from the new stream of digital pictures and from the first stream of digital pictures may be determined factoring in at least one of: an order of a digital picture in a stream of digital pictures, an importance of a digital picture, a threshold for a similarity in a digital picture, or a threshold for a difference in a digital picture. In further embodiments, the determining that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes determining that a number or a percentage of matching regions from at least one digital picture of the new stream of digital pictures and from at least one digital picture of the first stream of digital pictures exceeds a threshold number or threshold percentage. The matching regions from at least one digital picture of the new stream of digital pictures and from at least one digital picture of the first stream of digital pictures may be determined factoring in at least one of: a location of a region, an importance of a region, a threshold for a similarity in a region, or a threshold for a difference in a region. In further embodiments, the determining that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes determining that a number or a percentage of matching features from at least one digital picture of the new stream of digital pictures and from at least one digital picture of the first stream of digital pictures exceeds a threshold number or threshold percentage. The matching features from at least one digital picture of the new stream of digital pictures and from at least one digital picture of the first stream of digital pictures may be determined factoring in at least one of: a type of a feature, an importance of a feature, a location of a feature, a threshold for a similarity in a feature, or a threshold for a difference in a feature. In further embodiments, the determining that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes determining that a number or a percentage of matching pixels from at least one digital picture of the new stream of digital pictures and from at least one digital picture of the first stream of digital pictures exceeds a threshold number or threshold percentage. The matching pixels from at least one digital picture of the new stream of digital pictures and from at least one digital picture of the first stream of digital pictures may be determined factoring in at least one of: a location of a pixel, a threshold for a similarity in a pixel, or a threshold for a difference in a pixel. In further embodiments, the determining that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes recognizing a same person or object in the new and the first streams of digital pictures.

In certain embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures instead of or prior to an instruction set that would have been executed next. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying one or more instruction sets. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying a register or an element of a processor circuit. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes inserting the one or more instruction sets for operating the device correlated with the first stream of digital pictures into a register or an element of a processor circuit. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes redirecting a processor circuit to the one or more instruction sets for operating the device correlated with the first stream of digital pictures. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes redirecting a processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the one or more instruction sets for operating the device correlated with the first stream of digital pictures. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes transmitting, to a processor circuit for execution, the one or more instruction sets for operating the device correlated with the first stream of digital pictures. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes issuing an interrupt to a processor circuit and executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures following the interrupt. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying an element that is part of, operating on, or coupled to a processor circuit.

In some embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes executing, by a logic circuit, the one or more instruction sets for operating the device correlated with the first stream of digital pictures. The logic circuit may include a microcontroller. The executing, by the logic circuit, the one or more instruction sets for operating the device correlated with the first stream of digital pictures may include modifying an element of the logic circuit. The executing, by the logic circuit, the one or more instruction sets for operating the device correlated with the first stream of digital pictures may include inserting the one or more instruction sets for operating the device correlated with the first stream of digital pictures into an element of the logic circuit. The executing, by the logic circuit, the one or more instruction sets for operating the device correlated with the first stream of digital pictures may include redirecting the logic circuit to the one or more instruction sets for operating the device correlated with the first stream of digital pictures. The executing, by the logic circuit, the one or more instruction sets for operating the device correlated with the first stream of digital pictures may include replacing inputs into the logic circuit with the one or more instruction sets for operating the device correlated with the first stream of digital pictures. The executing, by the logic circuit, the one or more instruction sets for operating the device correlated with the first stream of digital pictures my include replacing

outputs from the logic circuit with the one or more instruction sets for operating the device correlated with the first stream of digital pictures.

In certain embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes executing, by an application for operating the device, the one or more instruction sets for operating the device correlated with the first stream of digital pictures. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying an application, the application including instruction sets for operating the device. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes redirecting an application to the one or more instruction sets for operating the device correlated with the first stream of digital pictures. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes redirecting an application to one or more alternate instruction sets, the alternate instruction sets comprising the one or more instruction sets for operating the device correlated with the first stream of digital pictures. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying one or more instruction sets of an application. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying at least one of: a memory unit, a register of a processor circuit, a storage, or a repository where instruction sets are stored or used. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying one or more instruction sets for operating an application or an object of the application. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying at least one of: an element of a processor circuit, an element of the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes modifying one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of an application. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream

of digital pictures includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes utilizing an assembly language. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes adding or inserting additional code into a code of an application. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes at least one of: modifying, removing, rewriting, or overwriting a code of an application. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes at least one of: branching, redirecting, extending, or hot swapping a code of an application. The branching or redirecting the code may include inserting at least one of: a branch, a jump, or a means for redirecting an execution. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes implementing a user's knowledge, style, or methodology of operating the device in a visual surrounding. In further embodiments, the executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures includes executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures via an interface. The interface may include a modification interface.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving at least one extra information. In further embodiments, the at least one extra information include one or more of: an information on a stream of digital pictures, an information on an object in the stream of digital pictures, an information on the device's visual surrounding, an information on an instruction set, an information on an application, an information on an object of the application, an information on a processor circuit, an information on the device, or an information on an user. In further embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: learning the first stream of digital pictures correlated with the at least one extra information. The learning the first stream of digital pictures correlated with at least one extra information may include correlating the first stream of digital pictures with the at least one extra information. The learning the first stream of digital pictures correlated with at least one extra information may include storing the first stream of digital pictures correlated with the at least one extra information into a memory unit. In further embodiments, the anticipating the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between the new stream of digital pictures and the first stream of digital pictures includes anticipating the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between an extra information correlated with the new stream of digital pictures and an extra information correlated with the first stream of digital

pictures. The anticipating the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between an extra information correlated with the new stream of digital pictures and an extra information correlated with the first stream of digital pictures may include comparing an extra information correlated with the new stream of digital pictures and an extra information correlated with the first stream of digital pictures. The anticipating the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between an extra information correlated with the new stream of digital pictures and an extra information correlated with the first stream of digital pictures may include determining that a similarity between an extra information correlated with the new stream of digital pictures and an extra information correlated with the first stream of digital pictures exceeds a similarity threshold.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving a second stream of digital pictures from the picture capturing apparatus; receiving additional one or more instruction sets for operating the device; and learning the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device. In further embodiments, the learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device and the learning the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device include creating a connection between the first stream of digital pictures correlated with the one or more instruction sets for operating the device and the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device. The connection may include or be associated with at least one of: an occurrence count, a weight, a parameter, or a data. In further embodiments, the learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device and the learning the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device include updating a connection between the first stream of digital pictures correlated with the one or more instruction sets for operating the device and the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device. The updating the connection between the first stream of digital pictures correlated with the one or more instruction sets for operating the device and the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device may include updating at least one of: an occurrence count, a weight, a parameter, or a data included in or associated with the connection. In further embodiments, the learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device includes storing the first stream of digital pictures correlated with the one or more instruction sets for operating the device into a first node of a data structure, and wherein the learning the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device includes storing the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device into a second node of the data structure. The data structure may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, or a knowledge structure. The learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device and the learning the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device may include creating a connection between the first node and the second node. The learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device and the learning the second stream of digital pictures correlated with the additional one or more instruction sets for

operating the device may include updating a connection between the first node and the second node. In further embodiments, the first stream of digital pictures correlated with the one or more instruction sets for operating the device is stored into a first node of a neural network and the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device is stored into a second node of the neural network. The first node and the second node may be connected by a connection. The first node may be part of a first layer of the neural network and the second node may be part of a second layer of the neural network. In further embodiments, the first stream of digital pictures correlated with the one or more instruction sets for operating the device is stored into a first node of a graph and the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device is stored into a second node of the graph. The first node and the second node may be connected by a connection. In further embodiments, the first stream of digital pictures correlated with the one or more instruction sets for operating the device is stored into a first node of a sequence and the second stream of digital pictures correlated with the additional one or more instruction sets for operating the device is stored into a second node of the sequence.

In some aspects, the disclosure relates to a system for learning a visual surrounding for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises a processor circuit configured to execute instruction sets for operating a device. The system may further include a memory unit configured to store data. The system may further include a picture capturing apparatus configured to capture digital pictures. The system may further include an artificial intelligence unit. In some embodiments, the artificial intelligence unit may be configured to: receive a first stream of digital pictures from the picture capturing apparatus. The artificial intelligence unit may be further configured to: receive one or more instruction sets for operating the device from the processor circuit. The artificial intelligence unit may be further configured to: learn the first stream of digital pictures correlated with the one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first stream of digital pictures from a picture capturing apparatus. The operations may further include: receiving one or more instruction sets for operating a device. The operations may further include: learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first stream of digital pictures from a picture capturing apparatus by one or more processor circuits. The method may further include: (b) receiving one or more instruction sets for operating a device by the one or more processor circuits. The method may further include: (c) learning the first stream of digital pictures correlated with the one or more instruction sets for operating the device, the learning of (c) performed by the one or more processor circuits.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems as applicable.

In some aspects, the disclosure relates to a system for using a visual surrounding for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises a processor circuit configured to execute instruction sets for operating a device. The system may further include a memory unit configured to store data. The system may further include a picture capturing apparatus configured to capture digital pictures. The system may further include an artificial intelligence unit. In some embodiments, the artificial intelligence unit may be configured to: access the memory unit that stores a plurality of streams of digital pictures correlated with one or more instruction sets for operating the device, the plurality including a first stream of digital pictures correlated with one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: receive a new stream of digital pictures from the picture capturing apparatus. The artificial intelligence unit may be further configured to: anticipate the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between the new stream of digital pictures and the first stream of digital pictures. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the one or more instruction sets for operating the device correlated with the first stream of digital pictures, the executing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the one or more instruction sets for operating the device correlated with the first stream of digital pictures, the one or more operations performed in response to the executing by the processor circuit.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: accessing a memory unit that stores a plurality of streams of digital pictures correlated with one or more instruction sets for operating a device, the plurality including a first stream of digital pictures correlated with one or more instruction sets for operating the device. The operations may further include: receiving a new stream of digital pictures from a picture capturing apparatus. The operations may further include: anticipating the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between the new stream of digital pictures and the first stream of digital pictures. The operations may further include: causing an execution of the one or more instruction sets for operating the device correlated with the first stream of digital pictures, the causing performed in response to the anticipating the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between the new stream of digital pictures and the first stream of digital pictures, wherein the device performs one or more operations defined by the one or more instruction sets for operating the device correlated with the first stream of digital pictures, the one or more operations performed in response to the executing.

In some aspects, the disclosure relates to a method comprising: (a) accessing a memory unit that stores a plurality of streams of digital pictures correlated with one or more instruction sets for operating a device, the plurality including a first stream of digital pictures correlated with one or more instruction sets for operating the device, the accessing of (a) performed by the one or more processor circuits. The method may further include: (b) receiving a new stream of digital pictures from a picture capturing apparatus by the one or more processor circuits. The method may further include: (c) anticipating the one or more instruction sets for operating the device correlated with the first stream of digital pictures based on at least a partial match between the new stream of digital pictures and the first

stream of digital pictures, the anticipating of (c) performed by the one or more processor circuits. The method may further include: (d) executing the one or more instruction sets for operating the device correlated with the first stream of digital pictures, the executing of (d) performed in response to the anticipating of (c). The method may further include: (e) performing, by the device, one or more operations defined by the one or more instruction sets for operating the device correlated with the first stream of digital pictures, the one or more operations performed in response to the executing of (d).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems as applicable.

In some aspects, the disclosure relates to a system for learning and using a visual surrounding for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises a logic circuit configured to receive inputs and produce outputs, the outputs for operating a device. The system may further include a memory unit configured to store data. The system may further include a picture capturing apparatus configured to capture digital pictures. The system may further include an artificial intelligence unit. In some embodiments, the artificial intelligence unit may be configured to: receive a first digital picture from the picture capturing apparatus. The artificial intelligence unit may be further configured to: receive at least one input, wherein the at least one input is also received by the logic circuit. The artificial intelligence unit may be further configured to: learn the first digital picture correlated with the at least one input. The artificial intelligence unit may be further configured to: receive a new digital picture from the picture capturing apparatus. The artificial intelligence unit may be further configured to: anticipate the at least one input correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture. The artificial intelligence unit may be further configured to: cause the logic circuit to receive the at least one input correlated with the first digital picture, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs at least one operation defined by at least one output for operating the device produced by the logic circuit.

In certain embodiments, the logic circuit configured to receive inputs and produce outputs includes a logic circuit configured to produce outputs based at least in part on logic operations performed on the inputs. In further embodiments, the learning the first digital picture correlated with the at least one input includes correlating the first digital picture with the at least one input. In further embodiments, the learning the first digital picture correlated with the at least one input includes storing, into the memory unit, the first digital picture correlated with the at least one input, the first digital picture correlated with the at least input being part of a stored plurality of digital pictures correlated with at least one input. In further embodiments, the anticipating the at least one input correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture includes comparing at least one portion of the new digital picture with at least one portion of the first digital picture. In further embodiments, the anticipating the at least one input correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture includes determining that there is at least a partial match between the new digital picture and the first digital picture. In further embodiments, the causing the logic circuit to receive the at least one input correlated with the first digital picture includes transmitting, to the

logic circuit, the at least one input correlated with the first digital picture. In further embodiments, the causing the logic circuit to receive the at least one input correlated with the first digital picture includes replacing at least one input into the logic circuit with the at least one input correlated with the first digital picture.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first digital picture from a picture capturing apparatus. The operations may further include: receiving at least one input, wherein the at least one input is also received by a logic circuit, and wherein the logic circuit is configured to receive inputs and produce outputs, the outputs for operating a device. The operations may further include: learning the first digital picture correlated with the at least one input. The operations may further include: receiving a new digital picture from the picture capturing apparatus. The operations may further include: anticipating the at least one input correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture. The operations may further include: causing the logic circuit to receive the at least one input correlated with the first digital picture, the causing performed in response to the anticipating the at least one input correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture, wherein the device performs at least one operation defined by at least one output for operating the device produced by the logic circuit.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first digital picture from a picture capturing apparatus by one or more processor circuits. The method may further include: (b) receiving at least one input by the one or more processor circuits, wherein the at least one input are also received by a logic circuit, and wherein the logic circuit is configured to receive inputs and produce outputs, the outputs for operating a device. The method may further include: (c) learning the first digital picture correlated with the at least one input, the learning of (c) performed by the one or more processor circuits. The method may further include: (d) receiving a new digital picture from the picture capturing apparatus by the one or more processor circuits. The method may further include: (e) anticipating the at least one input correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture, the anticipating of (e) performed by the one or more processor circuits. The method may further include: (f) receiving, by the logic circuit, the at least one input correlated with the first digital picture, the receiving of (f) performed in response to the anticipating of (e). The method may further include: (g) performing, by the device, at least one operation defined by at least one output for operating the device produced by the logic circuit.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems as applicable.

In some aspects, the disclosure relates to a system for learning and using a visual surrounding for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises a logic circuit configured to receive inputs and produce outputs, the outputs for operating a device. The system may further include a memory unit configured to store data. The system may further include a picture capturing apparatus configured to capture digital pictures. The system may further

include an artificial intelligence unit. In some embodiments, the artificial intelligence unit may be configured to: receive a first digital picture from the picture capturing apparatus. The artificial intelligence unit may be further configured to: receive at least one output, the at least one output transmitted from the logic circuit. The artificial intelligence unit may be further configured to: learn the first digital picture correlated with the at least one output. The artificial intelligence unit may be further configured to: receive a new digital picture from the picture capturing apparatus. The artificial intelligence unit may be further configured to: anticipate the at least one output correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture. The artificial intelligence unit may be further configured to: cause the device to perform at least one operation defined by the at least one output correlated with the first digital picture.

In certain embodiments, the logic circuit configured to receive inputs and produce outputs includes a logic circuit configured to produce outputs based at least in part on logic operations performed on the inputs. In further embodiments, the learning the first digital picture correlated with the at least one output includes correlating the first digital picture with the at least one output. In further embodiments, the learning the first digital picture correlated with the at least one output includes storing, into the memory unit, the first digital picture correlated with the at least one output, the first digital picture correlated with the at least output being part of a stored plurality of digital pictures correlated with at least one output. In further embodiments, the anticipating the at least one output correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture includes comparing at least one portion of the new digital picture with at least one portion of the first digital picture. In further embodiments, the anticipating the at least one output correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture includes determining that there is at least a partial match between the new digital picture and the first digital picture. In further embodiments, the causing the device to perform at least one operation defined by the at least one output correlated with the first digital picture includes replacing at least one output from the logic circuit with the at least one output correlated with the first digital picture.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first digital picture from the picture capturing apparatus. The operations may further include: receiving at least one output, the at least one output transmitted from a logic circuit, wherein the logic circuit is configured to receive inputs and produce outputs, the outputs for operating a device. The operations may further include: learning the first digital picture correlated with the at least one output. The operations may further include: receiving a new digital picture from the picture capturing apparatus. The operations may further include: anticipating the at least one output correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture. The operations may further include: causing the device to perform at least one operation defined by the at least one output correlated with the first digital picture.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first digital picture from the picture capturing apparatus by one or more processor circuits. The method may further include: (b) receiving at least one output by the one or more processor circuits, the at least one output transmitted from a logic circuit, wherein the logic circuit is configured to receive inputs and produce outputs, the outputs for operating a device. The method may

further include: (c) learning the first digital picture correlated with the at least one output, the learning of (c) performed by the one or more processor circuits. The method may further include: (d) receiving a new digital picture from the picture capturing apparatus by the one or more processor circuits. The method may further include: (e) anticipating the at least one output correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture, the anticipating of (e) performed by the one or more processor circuits. The method may further include: (f) performing, by the device, at least one operation defined by the at least one output correlated with the first digital picture.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems as applicable.

In some aspects, the disclosure relates to a system for learning and using a visual surrounding for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises an actuator configured to receive inputs and perform motions. The system may further include a memory unit configured to store data. The system may further include a picture capturing apparatus configured to capture digital pictures. The system may further include an artificial intelligence unit. In some embodiments, the artificial intelligence unit may be configured to: receive a first digital picture from the picture capturing apparatus. The artificial intelligence unit may be further configured to: receive at least one input, wherein the at least one input is also received by the actuator. The artificial intelligence unit may be further configured to: learn the first digital picture correlated with the at least one input. The artificial intelligence unit may be further configured to: receive a new digital picture from the picture capturing apparatus. The artificial intelligence unit may be further configured to. The artificial intelligence unit may be further configured to: anticipate the at least one input correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture. The artificial intelligence unit may be further configured to: cause the actuator to receive the at least one input correlated with the first digital picture, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the actuator performs at least one motion defined by the at least one input correlated with the first digital picture.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first digital picture from a picture capturing apparatus. The operations may further include: receiving at least one input, wherein the at least one input is also received by an actuator, and wherein the actuator is configured to receive inputs and perform motions. The operations may further include: learning the first digital picture correlated with the at least one input. The operations may further include: receiving a new digital picture from the picture capturing apparatus. The operations may further include: anticipating the at least one input correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture. The operations may further include: causing the actuator to receive the at least one input correlated with the first digital picture, the causing performed in response to the anticipating the at least one input correlated with the first digital picture based on at least a partial

match between the new digital picture and the first digital picture, wherein the actuator performs at least one motion defined by the at least one input correlated with the first digital picture.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first digital picture from a picture capturing apparatus by one or more processor circuits. The method may further include: (b) receiving at least one input by the one or more processor circuits, wherein the at least one input are also received by an actuator, and wherein the actuator is configured to receive inputs and perform motions. The method may further include: (c) learning the first digital picture correlated with the at least one input, the learning of (c) performed by the one or more processor circuits. The method may further include: (d) receiving a new digital picture from the picture capturing apparatus by the one or more processor circuits. The method may further include: (e) anticipating the at least one input correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture, the anticipating of (e) performed by the one or more processor circuits. The method may further include: (f) receiving, by the actuator, the at least one input correlated with the first digital picture, the receiving of (f) performed in response to the anticipating of (e). The method may further include: (g) performing, by the actuator, at least one motion defined by the at least one input correlated with the first digital picture.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems as applicable.

Other features and advantages of the disclosure will become apparent from the following description, including the claims and drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

Fig. 1 illustrates a block diagram of Computing Device 70 that can provide processing capabilities used in some of the disclosed embodiments.

Fig. 2 illustrates an embodiment of Device 98 comprising Unit for Learning and/or Using Visual Surrounding for Autonomous Device Operation (VSADO Unit 100).

Fig. 3 illustrates some embodiments of obtaining instruction sets, data, and/or other information through tracing, profiling, or sampling of Processor 11 registers, memory, or other computing system components.

Figs. 4A-4B illustrate some embodiments of obtaining instruction sets, data, and/or other information through tracing, profiling, or sampling of Logic Circuit 250.

Figs. 5A-5E illustrate some embodiments of Instruction Sets 526.

Figs. 6A-6B illustrate some embodiments of Extra Information 527.

Fig. 7 illustrates an embodiment where VSADO Unit 100 is part of or operating on Processor 11.

Fig. 8 illustrates an embodiment where VSADO Unit 100 resides on Server 96 accessible over Network 95.

Fig. 9 illustrates an embodiment where Picture Capturing Apparatus 90 is part of Remote Device 97 accessible over Network 95.

Fig. 10 illustrates an embodiment of VSADO Unit 100 comprising Picture Recognizer 350.

Fig. 11 illustrates an embodiment of Artificial Intelligence Unit 110.

Fig. 12 illustrates an embodiment of Knowledge Structuring Unit 520 correlating individual Digital Pictures 525 with any Instruction Sets 526 and/or Extra Info 527.

Fig. 13 illustrates another embodiment of Knowledge Structuring Unit 520 correlating individual Digital Pictures 525 with any Instruction Sets 526 and/or Extra Info 527.

Fig. 14 illustrates an embodiment of Knowledge Structuring Unit 520 correlating streams of Digital Pictures 525 with any Instruction Sets 526 and/or Extra Info 527.

Fig. 15 illustrates another embodiment of Knowledge Structuring Unit 520 correlating streams of Digital Pictures 525 with any Instruction Sets 526 and/or Extra Info 527.

Fig. 16 illustrates various artificial intelligence methods, systems, and/or models that can be utilized in VSADO Unit 100 embodiments.

Figs. 17A-17C illustrate embodiments of interconnected Knowledge Cells 800 and updating weights of Connections 853.

Fig. 18 illustrates an embodiment of learning Knowledge Cells 800 comprising one or more Digital Pictures 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Collection of Knowledge Cells 530d.

Fig. 19 illustrates an embodiment of learning Knowledge Cells 800 comprising one or more Digital Pictures 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Neural Network 530a.

Fig. 20 illustrates an embodiment of learning Knowledge Cells 800 comprising one or more Digital Pictures 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Neural Network 530a comprising shortcut Connections 853.

Fig. 21 illustrates an embodiment of learning Knowledge Cells 800 comprising one or more Digital Pictures 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Graph 530b.

Fig. 22 illustrates an embodiment of learning Knowledge Cells 800 comprising one or more Digital Pictures 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Collection of Sequences 530c.

Fig. 23 illustrates an embodiment of determining anticipatory Instruction Sets 526 from a single Knowledge Cell 800.

Fig. 24 illustrates an embodiment of determining anticipatory Instruction Sets 526 by traversing a single Knowledge Cell 800.

Fig. 25 illustrates an embodiment of determining anticipatory Instruction Sets 526 using collective similarity comparisons.

Fig. 26 illustrates an embodiment of determining anticipatory Instruction Sets 526 using Neural Network 530a.

Fig. 27 illustrates an embodiment of determining anticipatory Instruction Sets 526 using Graph 530b.

Fig. 28 illustrates an embodiment of determining anticipatory Instruction Sets 526 using Collection of Sequences 530c.

Fig. 29 illustrates some embodiments of modifying execution and/or functionality of Processor 11 through modification of Processor 11 registers, memory, or other computing system components.

Figs. 30A-30B illustrate some embodiments of modifying execution and/or functionality of Logic Circuit 250 through modification of inputs and/or outputs of Logic Circuit 250.

Fig. 31 illustrates a flow chart diagram of an embodiment of method 6100 for learning and/or using visual surrounding for autonomous device operation.

Fig. 32 illustrates a flow chart diagram of an embodiment of method 6200 for learning and/or using visual surrounding for autonomous device operation.

Fig. 33 illustrates a flow chart diagram of an embodiment of method 6300 for learning and/or using visual surrounding for autonomous device operation.

Fig. 34 illustrates a flow chart diagram of an embodiment of method 6400 for learning and/or using visual surrounding for autonomous device operation.

Fig. 35 illustrates a flow chart diagram of an embodiment of method 6500 for learning and/or using visual surrounding for autonomous device operation.

Fig. 36 illustrates a flow chart diagram of an embodiment of method 6600 for learning and/or using visual surrounding for autonomous device operation.

Fig. 37 illustrates an exemplary embodiment of Computing-enabled Machine 98a.

Fig. 38 illustrates an exemplary embodiment of Computing-enabled Machine 98a comprising or coupled to a plurality of Picture Capturing Apparatuses 90.

Fig. 39 illustrates an exemplary embodiment of Fixture 98b.

Fig. 40 illustrates an exemplary embodiment of Control Device 98c.

Fig. 41 illustrates an exemplary embodiment of Smartphone 98d.

Like reference numerals in different figures indicate like elements. Horizontal or vertical "…" or other such indicia may be used to indicate additional instances of the same type of element. n, m, x, or other such letters or indicia represent integers or other sequential numbers that follow the sequence where they are indicated. It should be noted that n, m, x, or other such letters or indicia may represent different numbers in different elements even where the elements are depicted in the same figure. In general, n, m, x, or other such letters or indicia may follow the sequence and/or context where they are indicated. Any of these or other such letters or indicia may be used interchangeably depending on the context and space available. The drawings are not necessarily to scale, with emphasis instead being placed upon illustrating the embodiments, principles, and concepts of the disclosure. A line or arrow between any of the disclosed elements comprises an interface that enables the coupling, connection, and/or interaction between the elements.

DETAILED DESCRIPTION

The disclosed artificially intelligent devices, systems, and methods for learning and/or using visual surrounding for autonomous device operation comprise apparatuses, systems, methods, features, functionalities, and/or applications that enable learning one or more digital pictures of a device's surrounding along with correlated instruction sets for operating the device, storing this knowledge in a knowledgebase (i.e. neural network, graph, sequences, etc.), and autonomously operating a device. The disclosed artificially intelligent devices, systems, and methods for learning and/or using visual surrounding for autonomous device operation, any of their elements, any of their embodiments, or a combination thereof can generally be referred to as VSADO, VSADO Unit, or as other similar name or reference.

Referring now to Fig. 1, an embodiment is illustrated of Computing Device 70 (also referred to simply as

computing device or other similar name or reference, etc.) that can provide processing capabilities used in some embodiments of the forthcoming disclosure. Later described devices and systems, in combination with processing capabilities of Computing Device 70, enable learning and/or using a device's visual surrounding for autonomous device operation and/or other functionalities described herein. Various embodiments of the disclosed devices, systems, and/or methods include hardware, functions, logic, programs, and/or a combination thereof that can be provided or implemented on any type or form of computing, computing enabled, or other device such as a mobile device, a computer, a computing enabled telephone, a server, a cloud device, a gaming device, a television device, a digital camera, a GPS receiver, a media player, an embedded device, a supercomputer, a wearable device, an implantable device, or any other type or form of computing, computing enabled, or other device capable of performing the operations described herein.

In some designs, Computing Device 70 comprises hardware, processing techniques or capabilities, programs, or a combination thereof. Computing Device 70 includes one or more central processing units, which may also be referred to as processors 11. Processor 11 includes one or more memory ports 10 and/or one or more input-output ports, also referred to as I/O ports 15, such as I/O ports 15A and 15B. Processor 11 may be special or general purpose. Computing Device 70 may further include memory 12, which can be connected to the remainder of the components of Computing Device 70 via bus 5. Memory 12 can be connected to processor 11 via memory port 10. Computing Device 70 may also include display device 21 such as a monitor, projector, glasses, and/or other display device. Computing Device 70 may also include Human-machine Interface 23 such as a keyboard, a pointing device, a mouse, a touchscreen, a joystick, and/or other input device that can be connected with the remainder of the Computing Device 70 components via I/O control 22. In some implementations, Human-machine Interface 23 can be connected with bus 5 or directly connected with specific components of Computing Device 70. Computing Device 70 may include additional elements, such as one or more input/output devices 13. Processor 11 may include or be interfaced with cache memory 14. Storage 27 may include memory, which provides an operating system, also referred to as OS 17, additional application programs 18 operating on OS 17, and/or data space 19 in which additional data or information can be stored. Alternative memory device 16 can be connected to the remaining components of Computing Device 70 via bus 5. Network interface 25 can also be connected with bus 5 and be used to communicate with external computing devices via a network. Some or all described elements of Computing Device 70 can be directly or operatively connected or coupled with each other using any other connection means known in art. Other additional elements may be included as needed, or some of the disclosed ones may be excluded, or a combination thereof may be utilized in alternate implementations of Computing Device 70.

Processor 11 includes any logic circuitry that can respond to or process instructions fetched from memory 12 or other element. Processor 11 may also include any combination of hardware and/or processing techniques or capabilities for implementing or executing logic functions or programs. Processor 11 may include a single core or a multi core processor. Processor 11 includes the functionality for loading operating system 17 and operating any application programs 18 thereon. In some embodiments, Processor 11 can be provided in a microprocessing or a processing unit, such as, for example, Snapdragon processor produced by Qualcomm Inc., processor by Intel Corporation of Mountain View, California, processor manufactured by Motorola Corporation of Schaumburg, Ill.; processor manufactured by Transmeta Corporation of Santa Clara, Calif.; the RS/6000 processor, processor manufactured by International Business Machines of White Plains, N.Y.; processor manufactured by Advanced

Micro Devices of Sunnyvale, California, or any computing unit for performing similar functions. In other embodiments, processor 11 can be provided in a graphics processing unit (GPU), visual processing unit (VPU), or other highly parallel processing unit or circuit such as, for example, nVidia GeForce line of GPUs, AMD Radeon line of GPUs, and/or others. Such GPUs or other highly parallel processing units may provide superior performance in processing operations on neural networks and/or other data structures. In further embodiments, processor 11 can be provided in a micro controller such as, for example, Texas instruments, Atmel, Microchip Technology, ARM, Silicon Labs, Intel, and/or other lines of micro controllers, and/or others. In further embodiments, processor 11 includes any circuit (i.e. logic circuit, etc.) or device for performing logic operations. Computing Device 70 can be based on one or more of the aforementioned or other processors capable of operating as described herein.

Memory 12 includes one or more memory chips capable of storing data and allowing any storage location to be accessed by processor 11 and/or other element. Examples of Memory 12 include static random access memory (SRAM), Flash memory, Burst SRAM or SynchBurst SRAM (BSRAM), Dynamic random access memory (DRAM), Fast Page Mode DRAM (FPM DRAM), Enhanced DRAM (EDRAM), Extended Data Output RAM (EDO RAM), Extended Data Output DRAM (EDO DRAM), Burst Extended Data Output DRAM (BEDO DRAM), Enhanced DRAM (EDRAM), synchronous DRAM (SDRAM), JEDEC SRAM, PC100 SDRAM, Double Data Rate SDRAM (DDR SDRAM), Enhanced SDRAM (ESDRAM), SyncLink DRAM (SLDRAM), Direct Rambus DRAM (DRDRAM), Ferroelectric RAM (FRAM), and/or others. Memory 12 can be based on any of the above described memory chips, or any other available memory chips capable of operating as described herein. In some embodiments, processor 11 can communicate with memory 12 via a system bus 5. In other embodiments, processor 11 can communicate directly with memory 12 via a memory port 10.

Processor 11 can communicate directly with cache memory 14 via a connection means such as a secondary bus which may also sometimes be referred to as a backside bus. In some embodiments, processor 11 can communicate with cache memory 14 using the system bus 5. Cache memory 14 may typically have a faster response time than main memory 12 and can include a type of memory which is considered faster than main memory 12, such as for example SRAM, BSRAM, or EDRAM. Cache memory includes any structure such as multilevel caches, for example. In some embodiments, processor 11 can communicate with one or more I/O devices 13 via a system bus 5. Various busses can be used to connect processor 11 to any of the I/O devices 13, such as a VESA VL bus, an ISA bus, an EISA bus, a MicroChannel Architecture (MCA) bus, a PCI bus, a PCI-X bus, a PCI-Express bus, a NuBus, and/or others. In some embodiments, processor 11 can communicate directly with I/O device 13 via HyperTransport, Rapid I/O, or InfiniBand. In further embodiments, local busses and direct communication can be mixed. For example, processor 11 can communicate with an I/O device 13 using a local interconnect bus and communicate with another I/O device 13 directly. Similar configurations can be used for any other components described herein.

Computing Device 70 may further include alternative memory such as a SD memory slot, a USB memory stick, an optical drive such as a CD-ROM drive, a CD-R/RW drive, a DVD-ROM drive or a BlueRay disc, a hard-drive, and/or any other device comprising non-volatile memory suitable for storing data or installing application programs. Computing Device 70 may further include a storage device 27 comprising any type or form of non-volatile memory for storing an operating system (OS) such as any type or form of Windows OS, Mac OS, Unix OS, Linux OS, Android OS, iPhone OS, mobile version of Windows OS, an embedded OS, or any other OS that can operate

on Computing Device 70. Computing Device 70 may also include application programs 18, and/or data space 19 for storing additional data or information. In some embodiments, alternative memory 16 can be used as or similar to storage device 27. Additionally, OS 17 and/or application programs 18 can be operable from a bootable medium, such as for example, a flash drive, a micro SD card, a bootable CD or DVD, and/or other bootable medium.

Application Program 18 (also referred to as program, computer program, application, script, code, or other similar name or refeerence) comprises instructions that can provide functionality when executed by processor 11. As such, Application Program 18 may be used to operate (i.e. perform operations on/with) or control a device or system. Application program 18 can be implemented in a high-level procedural or object-oriented programming language, or in a low-level machine or assembly language. Any language used can be compiled, interpreted, or otherwise translated into machine language. Application program 18 can be deployed in any form including as a stand-alone program or as a module, component, subroutine, or other unit suitable for use in a computing system. Application program 18 does not necessarily correspond to a file in a file system. A program can be stored in a portion of a file that may hold other programs or data, in a single file dedicated to the program, or in multiple files (i.e. files that store one or more modules, sub programs, or portions of code, etc.). Application Program 18 can be delivered in various forms such as, for example, executable file, library, script, plugin, addon, applet, interface, console application, web application, application service provider (ASP)-type application, operating system, and/or other forms. Application program 18 can be deployed to be executed on one computing device or on multiple computing devices (i.e. cloud, distributed, or parallel computing, etc.), or at one site or distributed across multiple sites interconnected by a communication network.

Network interface 25 can be utilized for interfacing Computing Device 70 with other devices via a network through a variety of connections including standard telephone lines, wired or wireless connections, LAN or WAN links (i.e. 802.11, T1, T3, 56 kb, X.25, etc.), broadband connections (i.e. ISDN, Frame Relay, ATM, etc.), or a combination thereof. Examples of networks include the Internet, an intranet, an extranet, a local area network (LAN), a wide area network (WAN), a personal area network (PAN), a home area network (HAN), a campus area network (CAN), a metropolitan area network (MAN), a global area network (GAN), a storage area network (SAN), virtual network, a virtual private network (VPN), Bluetooth network, a wireless network, a wireless LAN, a radio network, a HomePNA, a power line communication network, a G.hn network, an optical fiber network, an Ethernet network, an active networking network, a client-server network, a peer-to-peer network, a bus network, a star network, a ring network, a mesh network, a star-bus network, a tree network, a hierarchical topology network, and/or other networks. Network interface 25 may include a built-in network adapter, network interface card, PCMCIA network card, card bus network adapter, wireless network adapter, Bluetooth network adapter, WiFi network adapter, USB network adapter, modem, and/or any other device suitable for interfacing Computing Device 70 with any type of network capable of communication and/or operations described herein.

Still referring to Fig. 1, I/O devices 13 may be present in various shapes or forms in Computing Device 70. Examples of I/O device 13 capable of input include a joystick, a keyboard, a mouse, a trackpad, a trackpoint, a touchscreen, a trackball, a microphone, a drawing tablet, a glove, a tactile input device, a still or video camera, and/or other input device. Examples of I/O device 13 capable of output include a video display, a touchscreen, a projector, a glasses, a speaker, a tactile output device, and/or other output device. Examples of I/O device 13 capable of input and output include a disk drive, an optical storage device, a modem, a network card, and/or other

input/output device. I/O device 13 can be interfaced with processor 11 via an I/O port 15, for example. I/O device 13 can also be controlled by I/O control 22 in some implementations. I/O control 22 may control one or more I/O devices such as Human-machine Interface 23 (i.e. keyboard, pointing device, touchscreen, joystick, mouse, optical pen, etc.). I/O control 22 enables any type or form of a device such as, for example, a video camera or microphone to be interfaced with other components of Computing Device 70. Furthermore, I/O device 13 may also provide storage such as or similar to storage 27, and/or alternative memory such as or similar to alternative memory 16 in some implementations.

An output interface such as a graphical user interface, an acoustic output interface, a tactile output interface, any device driver (i.e. audio, video, or other driver), and/or other output interface or system can be utilized to process output from elements of Computing Device 70 for conveyance on an output device such as Display 21. In some aspects, Display 21 or other output device itself may include an output interface for processing output from elements of Computing Device 70. Further, an input interface such as a keyboard listener, a touchscreen listener, a mouse listener, any device driver (i.e. audio, video, keyboard, mouse, touchscreen, or other driver), a speech recognizer, a video interpreter, and/or other input interface or system can be utilized to process input from Human-machine Interface 23 or other input device for use by elements of Computing Device 70. In some aspects, Human-machine Interface 23 or other input device itself may include an input interface for processing input for use by elements of Computing Device 70.

Computing Device 70 may include or be connected to multiple display devices 21. Display devices 21 can each be of the same or different type or form. Computing Device 70 and/or its elements comprise any type or form of suitable hardware, programs, or a combination thereof to support, enable, or provide for the connection and use of multiple display devices 21. In one example, Computing Device 70 includes any type or form of video adapter, video card, driver, and/or library to interface, communicate, connect, or otherwise use display devices 21. In some aspects, a video adapter may include multiple connectors to interface to multiple display devices 21. In other aspects, Computing Device 70 includes multiple video adapters, with each video adapter connected to one or more display devices 21. In some embodiments, Computing Device's 70 operating system can be configured for using multiple displays 21. In other embodiments, one or more display devices 21 can be provided by one or more other computing devices such as remote computing devices connected to Computing Device 70 via a network.

In some embodiments, I/O device 13 can be a bridge between system bus 5 and an external communication bus, such as a USB bus, an Apple Desktop Bus, an RS-232 serial connection, a SCSI bus, a FireWire bus, a FireWire 800 bus, an Ethernet bus, an AppleTalk bus, a Gigabit Ethernet bus, an Asynchronous Transfer Mode bus, a HIPPI bus, a Super HIPPI bus, a SerialPlus bus, a SCI/LAMP bus, a FibreChannel bus, a Serial Attached small computer system interface bus, and/or other bus.

Computing Device 70 can operate under the control of operating system 17, which may support Computing Device's 70 basic functions, interface with and manage hardware resources, interface with and manage peripherals, provide common services for application programs, schedule tasks, and/or perform other functionalities. A modern operating system enables features and functionalities such as a high resolution display, graphical user interface (GUI), touchscreen, cellular network connectivity (i.e. mobile operating system, etc.), Bluetooth connectivity, WiFi connectivity, global positioning system (GPS) capabilities, mobile navigation, microphone, speaker, still picture camera, video camera, voice recorder, speech recognition, music player, video player, near field communication,

personal digital assistant (PDA), and/or other features, functionalities, or applications. For example, Computing Device 70 can use any conventional operating system, any embedded operating system, any real-time operating system, any open source operating system, any video gaming operating system, any proprietary operating system, any online operating system, any operating system for mobile computing devices, or any other operating system capable of running on Computing Device 70 and performing operations described herein. Example of operating systems include Windows XP, Windows 7, Windows 8, etc. manufactured by Microsoft Corporation of Redmond, Wash.; Mac OS, iPhone OS, etc. manufactured by Apple Computer of Cupertino, Calif.; OS/2 manufactured by International Business Machines of Armonk, N.Y.; Linux, a freely-available operating system distributed by Caldera Corp. of Salt Lake City, Utah; or any type or form of a Unix operating system, among others. Any operating systems such as the ones for Android devices can similarly be utilized.

Computing Device 70 can be implemented as or be part of various different model architectures such as web services, distributed computing, grid computing, cloud computing, and/or other architectures. For example, in addition to the traditional desktop, server, or mobile operating system architectures, a cloud-based operating system can be utilized to provide the structure on which embodiments of the disclosure can be implemented. Other aspects of Computing Device 70 can also be implemented in the cloud without departing from the spirit and scope of the disclosure. For example, memory, storage, processing, and/or other elements can be hosted in the cloud. In some embodiments, Computing Device 70 can be implemented on multiple devices. For example, a portion of Computing Device 70 can be implemented on a mobile device and another portion can be implemented on wearable electronics.

Computing Device 70 can be or include any mobile device, a mobile phone, a smartphone (i.e. iPhone, Windows phone, Blackberry, Android phone, etc.), a tablet, a personal digital assistant (PDA), wearable electronics, implantable electronics, or another mobile device capable of implementing the functionalities described herein. In other embodiments, Computing Device 70 can be or include an embedded device, which can be any device or system with a dedicated function within another device or system. Embedded systems range from the simplest ones dedicated to one task with no user interface to complex ones with advanced user interface that may resemble modern desktop computer systems. Examples of devices comprising an embedded device include a mobile telephone, a personal digital assistant (PDA), a gaming device, a media player, a digital still or video camera, a pager, a television device, a set-top box, a personal navigation device, a global positioning system (GPS) receiver, a portable storage device (i.e. a USB flash drive, etc.), a digital watch, a DVD player, a printer, a microwave oven, a washing machine, a dishwasher, a gateway, a router, a hub, an automobile entertainment system, an automobile navigation system, a refrigerator, a washing machine, a factory automation device, an assembly line device, a factory floor monitoring device, a thermostat, an automobile, a factory controller, a telephone, a network bridge, and/or other devices. An embedded device can operate under the control of an operating system for embedded devices such as MicroC/OS-II, QNX, VxWorks, eCos, TinyOS, Windows Embedded, Embedded Linux, and/or other embedded device operating systems.

Various implementations of the disclosed devices, systems, and/or methods can be realized in digital electronic circuitry, integrated circuitry, logic gates, specially designed application specific integrated circuits (ASICs), field programmable gate arrays (FPGAs), computer hardware, firmware, programs, virtual machines, and/or combinations thereof including their structural, logical, and/or physical equivalents.

The disclosed devices, systems, and/or methods may include clients and servers. A client and server are generally remote from each other and typically interact via a network. The relationship of a client and server may arise by virtue of computer programs running on their respective computers and having a client-server relationship to each other.

The disclosed devices, systems, and/or methods can be implemented in a computing system that includes a back end component, a middleware component, a front end component, or any combination thereof. The components of the system can be interconnected by any form or medium of digital data communication such as, for example, a network.

Computing Device 70 may include or be interfaced with a computer program product comprising instructions or logic encoded on a computer-readable medium. Such instructions or logic, when executed, may configure or cause a processor to perform the operations and/or functionalities disclosed herein. For example, a computer program can be provided or encoded on a computer-readable medium such as an optical medium (i.e. DVD-ROM, etc.), flash drive, hard drive, any memory, firmware, or other medium. Computer program can be installed onto a computing device to cause the computing device to perform the operations and/or functionalities disclosed herein. Machine-readable medium, computer-readable medium, or other such terms may refer to any computer program product, apparatus, and/or device for providing instructions and/or data to a programmable processor. As such, machine-readable medium includes any medium that can send or receive machine instructions as a machine-readable signal. Examples of a machine-readable medium include a volatile and/or non-volatile medium, a removable and/or non-removable medium, a communication medium, a storage medium, and/or other medium. A communication medium, for example, can transmit computer readable instructions and/or data in a modulated data signal such as a carrier wave or other transport technique, and may include any other form of information delivery medium known in art. A non-transitory machine-readable medium comprises all machine-readable media except for a transitory, propagating signal.

In some embodiments, the disclosed artificially intelligent devices, systems, and/or methods for learning and/or using visual surrounding for autonomous device operation, or elements thereof, can be implemented entirely or in part in a device (i.e. microchip, circuitry, logic gates, electronic device, computing device, special or general purpose processor, etc.) or system that comprises (i.e. hard coded, internally stored, etc.) or is provided with (i.e. externally stored, etc.) instructions for implementing VSADO functionalities. As such, the disclosed artificially intelligent devices, systems, and/or methods for learning and/or using visual surrounding for autonomous device operation, or elements thereof, may include the processing, memory, storage, and/or other features, functionalities, and embodiments of Computing Device 70 or elements thereof. Such device or system can operate on its own (i.e. standalone device or system, etc.), be embedded in another device or system (i.e. a television device, an oven, a refrigerator, a vehicle, an industrial machine, a robot, a smartphone, and/or any other device or system capable of housing the elements needed for VSADO functionalities), work in combination with other devices or systems, or be available in any other configuration. In other embodiments, the disclosed artificially intelligent devices, systems, and/or methods for learning and/or using visual surrounding for autonomous device operation, or elements thereof, may include Alternative Memory 16 that provides instructions for implementing VSADO functionalities to one or more Processors 11. In further embodiments, the disclosed artificially intelligent devices, systems, and/or methods for learning and/or using visual surrounding for autonomous device operation, or elements thereof, can be implemented

entirely or in part as a computer program and executed by one or more Processors 11. Such program can be implemented in one or more modules or units of a single or multiple computer programs. Such program may be able to attach to or interface with, inspect, and/or take control of another application program to implement VSADO functionalities. In further embodiments, the disclosed artificially intelligent devices, systems, and/or methods for learning and/or using visual surrounding for autonomous device operation, or elements thereof, can be implemented as a network, web, distributed, cloud, or other such application accessed on one or more remote computing devices (i.e. servers, cloud, etc.) via Network Interface 25, such remote computing devices including processing capabilities and instructions for implementing VSADO functionalities. In further embodiments, the disclosed artificially intelligent devices, systems, and/or methods for learning and/or using visual surrounding for autonomous device operation, or elements thereof, can be (1) attached to or interfaced with any computing device or application program, (2) included as a feature of an operating system, (3) built (i.e. hard coded, etc.) into any computing device or application program, and/or (4) available in any other configuration to provide its functionalities.

In yet other embodiments, the disclosed artificially intelligent devices, systems, and/or methods for learning and/or using visual surrounding for autonomous device operation, or elements thereof, can be implemented at least in part in a computer program such as Java application or program. Java provides a robust and flexible environment for application programs including flexible user interfaces, robust security, built-in network protocols, powerful application programming interfaces, database or DBMS connectivity and interfacing functionalities, file manipulation capabilities, support for networked applications, and/or other features or functionalities. Application programs based on Java can be portable across many devices, yet leverage each device's native capabilities. Java supports the feature sets of most smartphones and a broad range of connected devices while still fitting within their resource constraints. Various Java platforms include virtual machine features comprising a runtime environment for application programs. Java platforms provide a wide range of user-level functionalities that can be implemented in application programs such as displaying text and graphics, playing and recording audio content, displaying and recording visual content, communicating with another computing device, and/or other functionalities. It should be understood that the disclosed artificially intelligent devices, systems, and/or methods for learning and/or using visual surrounding for autonomous device operation, or elements thereof, are programming language, platform, and operating system independent. Examples of programming languages that can be used instead of or in addition to Java include C, C++, Cobol, Python, Java Script, Tcl, Visual Basic, Pascal, VB Script, Perl, PHP, Ruby, and/or other programming languages capable of implementing the functionalities described herein.

Where a reference to a specific file or file type is used herein, other files, file types, or formats can be substituted.

Where a reference to a data structure is used herein, it should be understood that any variety of data structures can be used such as, for example, array, list, linked list, doubly linked list, queue, tree, heap, graph, map, grid, matrix, multi-dimensional matrix, table, database, database management system (DBMS), file, neural network, and/or any other type or form of a data structure including a custom one. A data structure may include one or more fields or data fields that are part of or associated with the data structure. A field or data field may include a data, an object, a data structure, and/or any other element or a reference/pointer thereto. A data structure can be stored in one or more memories, files, or other repositories. A data structure and/or any elements thereof, when stored in a memory, file, or other repository, may be stored in a different arrangement than the arrangement of the data

structure and/or any elements thereof. For example, a sequence of elements can be stored in an arrangement other than a sequence in a memory, file, or other repository.

Where a reference to a repository is used herein, it should be understood that a repository may be or include one or more files or file systems, one or more storage locations or structures, one or more storage systems, one or more data structures or objects, one or more memory locations or structures, and/or other storage, memory, or data arrangements.

Where a reference to an interface is used herein, it should be understood that the interface comprises any hardware, device, system, program, method, and/or combination thereof that enable direct or operative coupling, connection, and/or interaction of the elements between which the interface is indicated. A line or arrow shown in the figures between any of the depicted elements comprises such interface. Examples of an interface include a direct connection, an operative connection, a wired connection (i.e. wire, cable, etc.), a wireless connection, a device, a network, a bus, a circuit, a firmware, a driver, a bridge, a program, a combination thereof, and/or others.

Where a reference to an element coupled or connected to another element is used herein, it should be understood that the element may be in communication or any other interactive relationship with the other element. Furthermore, an element coupled or connected to another element can be coupled or connected to any other element in alternate implementations. Terms coupled, connected, interfaced, or other such terms may be used interchangeably herein depending on context.

Where a reference to an element matching another element is used herein, it should be understood that the element may be equivalent or similar to the other element. Therefore, the term match or matching can refer to total equivalence or similarity depending on context.

Where a reference to a device is used herein, it should be understood that the device may include or be referred to as a system, and vice versa depending on context, since a device may include a system of elements and a system may be embodied in a device.

Where a mention of a function, method, routine, subroutine, or other such procedure is used herein, it should be understood that the function, method, routine, subroutine, or other such procedure comprises a call, reference, or pointer to the function, method, routine, subroutine, or other such procedure.

Where a mention of data, object, data structure, item, element, or thing is used herein, it should be understood that the data, object, data structure, item, element, or thing comprises a reference or pointer to the data, object, data structure, item, element, or thing.

The term collection of elements can refer to plurality of elements without implying that the collection is an element itself.

Referring to Fig. 2, an embodiment of Device 98 comprising Unit for Learning and/or Using Visual Surrounding for Autonomous Device Operation (VSADO Unit 100) is illustrated. Device 98 also comprises interconnected Processor 11, Human-machine Interface 23, Picture Capturing Apparatus 90, Memory 12, and Storage 27. Processor 11 includes or executes Application Program 18. VSADO Unit 100 comprises interconnected Artificial Intelligence Unit 110, Acquisition Interface 120, and Modification Interface 130. Other additional elements can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate embodiments.

In one example, the teaching presented by the disclosure can be implemented in a device or system for learning and/or using visual surrounding for autonomous device operation. The device or system may include a processor circuit (i.e. Processor 11, etc.) configured to execute instruction sets (i.e. Instruction Sets 526, etc.) for operating a device. The device or system may further include a memory unit (i.e. Memory 12, etc.) configured to store data. The device or system may further include a picture capturing apparatus (i.e. Picture Capturing Apparatus 90, etc.) configured to capture digital pictures (i.e. Digital Pictures 525, etc.). The device or system may further include an artificial intelligence unit (i.e. Artificial Intelligence Unit 110, etc.). The artificial intelligence unit may be configured to receive a first digital picture from the picture capturing apparatus. The artificial intelligence unit may also be configured to receive one or more instruction sets for operating the device from the processor circuit. The artificial intelligence unit may also be configured to learn the first digital picture correlated with the one or more instruction sets for operating the device. The artificial intelligence unit may also be configured to receive a new digital picture from the picture capturing apparatus. The artificial intelligence unit may also be configured to anticipate the one or more instruction sets for operating the device correlated with the first digital picture based on at least a partial match between the new digital picture and the first digital picture. The artificial intelligence unit may also be configured to cause the processor circuit to execute the one or more instruction sets for operating the device correlated with the first digital picture, the executing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the one or more instruction sets for operating the device correlated with the first digital picture, the one or more operations performed in response to the executing by the processor circuit. Any of the operations of the described elements can be performed repeatedly and/or in different orders in alternate embodiments. In some embodiments, a stream of digital pictures can be used instead of or in addition to any digital picture such as, for example, using a first stream of digital pictures instead of the first digital picture. In other embodiments, a logic circuit (i.e. Logic Circuit 250, etc.) may be used instead of the processor circuit. In such embodiments, the one or more instruction sets for operating the device may include or be substituted with one or more inputs into or one or more outputs from the logic circuit. In further embodiments, an actuator may be included instead of or in addition to the processor circuit. In such embodiments, the one or more instruction sets for operating the device may include or be substituted with one or more inputs into the actuator. Other additional elements can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate embodiments. The device or system for learning and/or using visual surrounding for autonomous device operation may include any actions or operations of any of the disclosed methods such as methods 6100, 6200, 6300, 6400, 6500, and/or 6600 (all later described).

Device 98 comprises any hardware, programs, or a combination thereof. Device 98 may include a system. Device 98 may include any features, functionalities, and embodiments of Computing Device 70, or elements thereof. Examples of Device 98 include a desktop or other computer, a smartphone or other mobile computer, a vehicle, an industrial machine, a toy, a robot, a microwave or other oven, and/or any other device or machine comprising processing capabilities. Such device or machine may be built for any function or purpose examples of which are described later.

User 50 (also referred to simply as user or other similar name or reference) comprises a human user or non-human user. A non-human User 50 includes any device, system, program, and/or other mechanism for operating or controlling Device 98 and/or elements thereof. In one example, User 50 may issue an operating

direction to Application Program 18 responsive to which Application Program's 18 instructions or instruction sets may be executed by Processor 11 to perform a desired operation on Device 98. In another example, User 50 may issue an operating direction to Processor 11, Logic Circuit 250 (later described), and/or other processing element responsive to which Processor 11, Logic Circuit 250, and/or other processing element may implement logic to perform a desired operation on Device 98. User's 50 operating directions comprise any user inputted data (i.e. values, text, symbols, etc.), directions (i.e. move right, move up, move forward, copy an item, click on a link, etc.), instructions or instruction sets (i.e. manually inputted instructions or instruction sets, etc.), and/or other inputs or information. A non-human User 50 can utilize more suitable interfaces instead of, or in addition to, Human-machine Interface 23 and/or Display 21 for controlling Device 98 and/or elements thereof. Examples of such interfaces include an application programming interface (API), bridge (i.e. bridge between applications, devices, or systems, etc.), driver, socket, direct or operative connection, handle, function/routine/subroutine, and/or other interfaces.

In some embodiments, Processor 11, Logic Circuit 250, Application Program 18, and/or other processing element may control or affect an actuator (not shown). Actuator comprises the functionality for implementing movements, actions, behaviors, maneuvers, and/or other mechanical or physical operations. Device 98 may include one or more actuators to enable Device 98 to perform mechanical, physical, or other operations and/or to interact with its environment. For example, an actuator can be connected to or coupled to an element such as a wheel, arm, or other element to act upon the environment. Examples of an actuator include a motor, a linear motor, a servomotor, a hydraulic element, a pneumatic element, an electro-magnetic element, a spring element, and/or other actuators. Examples of types of actuators include a rotary actuator, a linear actuator, and/or other types of actuators. In other embodiments, Processor 11, Logic Circuit 250, Application Program 18, and/or other processing element may control or affect any other device or element instead of or in addition to an actuator.

Picture Capturing Apparatus 90 comprises the functionality for capturing one or more pictures, and/or other functionalities. As such, Picture Capturing Apparatus 90 can be used to capture pictures of Device's 98 surrounding. In some embodiments, Picture Capturing Apparatus 90 may be or comprises a motion picture camera that can capture streams of pictures (i.e. motion pictures, videos, etc.). In other embodiments, Picture Capturing Apparatus 90 may be or comprises a still picture camera that can capture still pictures (i.e. photographs, etc.). In further embodiments, Picture Capturing Apparatus 90 may be or comprises any other picture capturing apparatus. In general, Picture Capturing Apparatus 90 may capture any light (i.e. visible light, infrared light, ultraviolet light, x-ray light, etc.) across the electromagnetic spectrum onto a light-sensitive material. In one example, a digital Picture Capturing Apparatus 90 can utilize a charge coupled device (CCD), a CMOS sensor, and/or other electronic image sensor to capture digital pictures that can then be stored in a memory or storage, or transmitted to an element such as Artificial Intelligence Unit 110. In another example, analog Picture Capturing Apparatus 90 can utilize an analog-to-digital converter to produce digital pictures. In some embodiments, Picture Capturing Apparatus 90 can be built, embedded, or integrated in Device 98, VSADO Unit 100, and/or other disclosed element. In other embodiments, Picture Capturing Apparatus 90 can be an external Picture Capturing Apparatus 90 connected with Device 98, VSADO Unit 100, and/or other disclosed element. In further embodiments, Picture Capturing Apparatus 90 comprises Computing Device 70 or elements thereof. In general, Picture Capturing Apparatus 90 can be implemented in any suitable configuration to provide its functionalities. Picture Capturing Apparatus 90 may capture one or more Digital Pictures 525. Digital Picture 525 (also referred to simply as digital pictures, etc.) may include a

collection of color encoded pixels or dots. Examples of file formats that can be utilized to store Digital Picture 525 include JPEG, GIF, TIFF, PNG, PDF, and/or other file formats. A stream of Digital Pictures 525 (i.e. motion picture, video, etc.) may include one or more Digital Pictures 525. Examples of file formats that can be utilized to store a stream of Digital Pictures 525 include MPEG, AVI, FLV, MOV, RM, SWF, WMV, DivX, and/or other file formats. In some aspects, Digital Picture 525 may include or be substituted with a stream of Digital Pictures 525, and vice versa. Therefore, the terms digital picture and stream of digital pictures may be used interchangeably herein depending on context. In some aspects, Device's 98 surrounding may include exterior of Device 98. In other aspects, Device's 98 surrounding may include interior of Device 98 in case of hollow Device 98, Device 98 comprising compartments or openings, and/or other variously shaped Device 98.

VSADO Unit 100 comprises any hardware, programs, or a combination thereof. VSADO Unit 100 comprises the functionality for learning the operation of Device 98 in various visual surroundings. VSADO Unit 100 comprises the functionality for structuring and/or storing this knowledge in a knowledgebase (i.e. neural network, graph, sequences, other repository, etc.). VSADO Unit 100 comprises the functionality for enabling autonomous operation of Device 98 in various visual surroundings. VSADO Unit 100 comprises the functionality for interfacing with or attaching to Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. VSADO Unit 100 comprises the functionality for obtaining instruction sets, data, and/or other information used, implemented, and/or executed by Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. VSADO Unit 100 comprises the functionality for modifying instruction sets, data, and/or other information used, implemented, and/or executed by Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. VSADO Unit 100 comprises learning, anticipating, decision making, automation, and/or other functionalities disclosed herein. Statistical, artificial intelligence, machine learning, and/or other models or techniques are utilized to implement the disclosed devices, systems, and methods.

When the disclosed VSADO Unit 100 functionalities are applied on Application Program 18, Processor 11, Logic Circuit 250 (later described), and/or other processing element of Device 98, Device 98 may become autonomous. VSADO Unit 100 may take control from, share control with, and/or release control to Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element to implement autonomous operation of Device 98. VSADO Unit 100 may take control from, share control with, and/or release control to Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element automatically or after prompting User 50 to allow it. In some aspects, Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element of an autonomous Device 98 may include or be provided with anticipatory instructions or instruction sets that User 50 did not issue or cause to be executed. Such anticipatory instructions or instruction sets include instruction sets that User 50 may want or is likely to issue or cause to be executed. Anticipatory instructions or instruction sets can be generated by VSADO Unit 100 or elements thereof based on the visual surrounding of Device 98. As such, Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element of an autonomous Device 98 may include or be provided with some or all original instructions or instruction sets and/or any anticipatory instructions or instruction sets generated by VSADO Unit 100. Therefore, autonomous Device 98 operating may include executing some or all original instructions or instruction sets and/or any anticipatory instructions or instruction sets generated by VSADO Unit 100. In one example, VSADO Unit 100 can overwrite or rewrite the original instructions or instruction sets of Application Program 18, Processor 11, Logic Circuit 250, and/or

other processing element with VSADO Unit 100-generated instructions or instruction sets. In another example, VSADO Unit 100 can insert or embed VSADO Unit 100-generated instructions or instruction sets among the original instructions or instruction sets of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. In a further example, VSADO Unit 100 can branch, redirect, or jump to VSADO Unit 100-generated instructions or instruction sets from the original instructions or instruction sets of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element.

In some embodiments, autonomous Device 98 operating comprises determining, by VSADO Unit 100, a next instruction or instruction set to be executed based on Device's 98 visual surrounding prior to the user issuing or causing to be executed the next instruction or instruction set. In yet other embodiments, autonomous application operating comprises determining, by VSADO Unit 100, a next instruction or instruction set to be executed based on Device's 98 visual surrounding prior to the system receiving the next instruction or instruction set.

In some embodiments, autonomous Device 98 operating includes a partially or fully autonomous operating. In an example involving partially autonomous Device 98 operating, a user confirms VSADO Unit 100-generated instructions or instruction sets prior to their execution. In an example involving fully autonomous application operating, VSADO Unit 100-generated instructions or instruction sets are executed without user or other system confirmation (i.e. automatically, etc.).

In some embodiments, a combination of VSADO Unit 100 and other systems and/or techniques can be utilized to implement Device's 98 operation. In one example, VSADO Unit 100 may be a primary or preferred system for implementing Device's 98 operation. While operating autonomously under the control of VSADO Unit 100, Device 98 may encounter a visual surrounding that has not been encountered or learned before. In such situations, User 50 and/or non-VSADO system may take control of Device's 98 operation. VSADO Unit 100 may take control again when Device 98 encounters a previously learned visual surrounding. Naturally, VSADO Unit 100 can learn Device's 98 operation in visual surroundings while User 50 and/or non-VSADO system is in control of Device 98, thereby reducing or eliminating the need for future involvement of User 50 and/or non-VSADO system. In another example, User 50 and/or non-VSADO system may be a primary or preferred system for control of Device's 98 operation. While operating under the control of User 50 and/or non-VSADO system, User 50 and/or non-VSADO system may release control to VSADO Unit 100 for any reason (i.e. User 50 gets tired or distracted, non-VSADO system gets stuck or cannot make a decision, etc.), at which point Device 98 can be controlled by VSADO Unit 100. In some designs, VSADO Unit 100 may take control in certain special visual surroundings where VSADO Unit 100 may offer superior performance even though User 50 and/or non-VSADO system may generally be preferred. Once Device 98 leaves such special visual surrounding, VSADO Unit 100 may release control to User 50 and/or a non-VSADO system. In general, VSADO Unit 100 can take control from, share control with, or release control to User 50, non-VSADO system, and/or other system or process at any time, under any circumstances, and remain in control for any period of time as needed.

In some embodiments, VSADO Unit 100 may control one or more sub-devices, sub-systems, or elements of Device 98 while User 50 and/or non-VSADO system may control other one or more sub-devices, sub-systems, or elements of Device 98.

It should be understood that a reference to autonomous operating of Device 98 may include autonomous operating of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element depending

on context.

Acquisition Interface 120 comprises the functionality for obtaining or receiving instruction sets, data, and/or other information. Acquisition Interface 120 comprises the functionality for obtaining or receiving instruction sets, data, and/or other information from Processor 11, Application Program 18, Logic Circuit 250 (later described), and/or other processing element. Acquisition Interface 120 comprises the functionality for obtaining or receiving instruction sets, data, and/or other information at runtime. In some aspects, an instruction set may include any computer command, instruction, signal, or input used in Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. Therefore, the terms instruction set, command, instruction, signal, input, or other such terms may be used interchangeably herein depending on context. Acquisition Interface 120 also comprises the functionality for attaching to or interfacing with Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. In one example, Acquisition Interface 120 comprises the functionality to access and/or read runtime engine/environment, virtual machine, operating system, compiler, just-in-time (JIT) compiler, interpreter, translator, execution stack, file, object, data structure, and/or other computing system elements. In another example, Acquisition Interface 120 comprises the functionality to access and/or read memory, storage, bus, interfaces, and/or other computing system elements. In a further example, Acquisition Interface 120 comprises the functionality to access and/or read Processor 11 registers and/or other Processor 11 elements. In a further example, Acquisition Interface 120 comprises the functionality to access and/or read inputs and/or outputs of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. In a further example, Acquisition Interface 120 comprises the functionality to access and/or read functions, methods, procedures, routines, subroutines, and/or other elements of Application Program 18. In a further example, Acquisition Interface 120 comprises the functionality to access and/or read source code, bytecode, compiled, interpreted, or otherwise translated code, machine code, and/or other code. In a further example, Acquisition Interface 120 comprises the functionality to access and/or read values, variables, parameters, and/or other data or information. Acquisition Interface 120 also comprises the functionality for transmitting the obtained instruction sets, data, and/or other information to Artificial Intelligence Unit 110 and/or other element. As such, Acquisition Interface 120 provides input into Artificial Intelligence Unit 110 for knowledge structuring, anticipating, decision making, and/or other functionalities later in the process. Acquisition Interface 120 also comprises other disclosed functionalities.

Acquisition Interface 120 can employ various techniques for obtaining instruction sets, data, and/or other information. In one example, Acquisition Interface 120 can attach to and/or obtain Processor's 11, Application Program's 18, Logic Circuit's 250, and/or other processing element's instruction sets, data, and/or other information through tracing or profiling techniques. Tracing or profiling may be used for outputting Processor's 11, Application Program's 18, Logic Circuit's 250, and/or other processing element's instruction sets, data, and/or other information at runtime. For instance, tracing or profiling may include adding trace code (i.e. instrumentation, etc.) to an application and/or outputting trace information to a specific target. The outputted trace information (i.e. instruction sets, data, and/or other information, etc.) can then be provided to or recorded into a file, data structure, repository, an application, and/or other system or target that may receive such trace information. As such, Acquisition Interface 120 can utilize tracing or profiling to obtain instruction sets, data, and/or other information and provide them as input into Artificial Intelligence Unit 110. In some aspects, instrumentation can be performed in source code, bytecode, compiled, interpreted, or otherwise translated code, machine code, and/or other code. In other aspects,

instrumentation can be performed in various elements of a computing system such as memory, virtual machine, runtime engine/environment, operating system, compiler, interpreter, translator, processor registers, execution stack, program counter, and/or other elements. In yet other aspects, instrumentation can be performed in various abstraction layers of a computing system such as in software layer (i.e. Application Program 18, etc.), in virtual machine (if VM is used), in operating system, in Processor 11, and/or in other layers or areas that may exist in a particular computing system implementation. In yet other aspects, instrumentation can be performed at various time periods in an application's execution such as source code write time, compile time, interpretation time, translation time, linking time, loading time, runtime, and/or other time periods. In yet other aspects, instrumentation can be performed at various granularities or code segments such as some or all lines of code, some or all statements, some or all instructions or instruction sets, some or all basic blocks, some or all functions/routines/subroutines, and/or some or all other code segments.

In some embodiments, Application Program 18 can be automatically instrumented. In one example, Acquisition Interface 120 can access Application Program's 18 source code, bytecode, or machine code and select instrumentation points of interest. Selecting instrumentation points may include finding locations in the source code, bytecode, or machine code corresponding to function calls, function entries, function exits, object creations, object destructions, event handler calls, new lines (i.e. to instrument all lines of code, etc.), thread creations, throws, and/or other points of interest. Instrumentation code can then be inserted at the instrumentation points of interest to output Application Program's 18 instruction sets, data, and/or other information. In response to executing instrumentation code, Application Program's 18 instruction sets, data, and/or other information may be received by Acquisition Interface 120. In some aspects, Application Program's 18 source code, bytecode, or machine code can be dynamically instrumented. For example, instrumentation code can be dynamically inserted into Application Program 18 at runtime.

In other embodiments, Application Program 18 can be manually instrumented. In one example, a programmer can instrument a function call by placing an instrumenting instruction immediately after the function call as in the following example.

```
Object1.moveRight(73);
traceApplication('Object1.moveRight(73);');
```

In another example, an instrumenting instruction can be placed immediately before the function call, or at the beginning, end, or anywhere within the function itself. A programmer may instrument all function calls or only function calls of interest. In a further example, a programmer can instrument all lines of code or only code lines of interest. In a further example, a programmer can instrument other elements utilized or implemented within Application Program 18 such as objects and/or any of their functions, data structures and/or any of their functions, event handlers and/or any of their functions, threads and/or any of their functions, and/or other elements or functions. Similar instrumentation as in the preceding examples can be performed automatically or dynamically. In some designs where manual code instrumentation is utilized, Acquisition Interface 120 can optionally be omitted and Application Program's 18 instruction sets, data, and/or other information may be transmitted directly to Artificial Intelligence Unit 110.

In some embodiments, VSADO Unit 100 can be selective in learning instruction sets, data, and/or other information to those implemented, utilized, or related to an object, data structure, repository, thread, function, and/or

other element of Application Program 18. In some aspects, Acquisition Interface 120 can obtain Application Program's 18 instruction sets, data, and/or other information implemented, utilized, or related to a certain object in an object oriented Application Program 18.

In some embodiments, various computing systems and/or platforms may provide native tools for obtaining instruction sets, data, and/or other information. Also, independent vendors may provide portable tools with similar functionalities that can be utilized across different computing systems and/or platforms. These native and portable tools may provide a wide range of functionalities to obtain runtime and other information such as instrumentation, tracing or profiling, logging application or system messages, outputting custom text messages, outputting objects or data structures, outputting functions/routines/subroutines or their invocations, outputting variable or parameter values, outputting thread or process behaviors, outputting call or other stacks, outputting processor registers, providing runtime memory access, providing inputs and/or outputs, performing live application monitoring, and/or other capabilities. One of ordinary skill in art will understand that, while all possible variations of the techniques to obtain instruction sets, data, and/or other information are too voluminous to describe, these techniques are within the scope of this disclosure.

In one example, obtaining instruction sets, data, and/or other information can be implemented through the .NET platform's native tools for application tracing or profiling such as System.Diagnostics.Trace, System.Diagnostics.Debug, and System.Diagnostics.TraceSource classes for tracing execution flow, and System.Diagnostics.Process, System.Diagnostics.EventLog, and System.Diagnostics.PerformanceCounter classes for profiling code, accessing local and remote processes, starting and stopping system processes, and interacting with Windows event logs, etc. For instance, a set of trace switches can be created that output an application's information. The switches can be configured using the .config file. For a Web application, this may typically be Web.config file associated with the project. In a Windows application, this file may typically be named applicationName.exe.config. Trace code can be added to application code automatically or manually as previously described. Appropriate listener can be created where the trace output is received. Trace code may output trace messages to a specific target such as a file, a log, a database, an object, a data structure, and/or other repository or system. Acquisition Interface 120 or Artificial Intelligence Unit 110 can then read or obtain the trace information from these targets. In some aspects, trace code may output trace messages directly to Acquisition Interface 120. In other aspects, trace code may output trace messages directly to Artificial Intelligence Unit 110. In the case of outputting trace messages to Acquisition Interface 120 or directly to Artificial Intelligence Unit 110, custom listeners can be built to accommodate these specific targets. Other platforms, tools, and/or techniques can provide equivalent or similar functionalities as the above described ones.

In another example, obtaining instruction sets, data, and/or other information can be implemented through the .NET platform's Profiling API that can be used to create a custom profiler application for tracing, monitoring, interfacing with, and/or managing a profiled application. The Profiling API provides an interface that includes methods to notify the profiler of events in the profiled application. The Profiling API may also provide an interface to enable the profiler to call back into the profiled application to obtain information about the state of the profiled application. The Profiling API may further provide call stack profiling functionalities. Call stack (also referred to as execution stack, control stack, runtime stack, machine stack, the stack, etc.) includes a data structure that can store information about active subroutines of an application. The Profiling API may provide a stack snapshot method,

which enables a trace of the stack at a particular point in time. The Profiling API may also provide a shadow stack method, which tracks the call stack at every instant. A shadow stack can obtain function arguments, return values, and information about generic instantiations. A function such as FunctionEnter can be utilized to notify the profiler that control is being passed to a function and can provide information about the stack frame and function arguments. A function such as FunctionLeave can be utilized to notify the profiler that a function is about to return to the caller and can provide information about the stack frame and function return value. An alternative to call stack profiling includes call stack sampling in which the profiler can periodically examine the stack. In some aspects, the Profiling API enables the profiler to change the in-memory code stream for a routine before it is just-in-time (JIT) compiled where the profiler can dynamically add instrumentation code to all or particular routines of interest. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through Java platform's APIs for application tracing or profiling such as Java Virtual Machine Profiling Interface (JVMPI), Java Virtual Machine Tool Interface (JVMTI), and/or other APIs or tools. These APIs can be used for instrumentation of an application, for notification of Java Virtual Machine (VM) events, and/or other functionalities. One of the tracing or profiling techniques that can be utilized includes bytecode instrumentation. The profiler can insert bytecodes into all or some of the classes. In application execution profiling, for example, these bytecodes may include methodEntry and methodExit calls. In memory profiling, for example, the bytecodes may be inserted after each new or after each constructor. In some aspects, insertion of instrumentation bytecode can be performed either by a post-compiler or a custom class loader. An alternative to bytecode instrumentation includes monitoring events generated by the JVMPI or JVMTI interfaces. Both APIs can generate events for method entry/exit, object allocation, and/or other events. In some aspects, JVMTI can be utilized for dynamic bytecode instrumentation where insertion of instrumentation bytecodes is performed at runtime. The profiler may insert the necessary instrumentation when a selected class is invoked in an application. This can be accomplished using the JVMTI's redefineClasses method, for example. This approach also enables changing of the level of profiling as the application is running. If needed, these changes can be made adaptively without restarting the application. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through JVMTI's programming interface that enables creation of software agents that can monitor and control a Java application. An agent may use the functionality of the interface to register for notification of events as they occur in the application, and to query and control the application. A JVMTI agent may use JVMTI functions to extract information from a Java application. A JVMTI agent can be utilized to obtain an application's runtime information such as method calls, memory allocation, CPU utilization, lock contention, and/or other information. JVMTI may include functions to obtain information about variables, fields, methods, classes, and/or other information. JVMTI may also provide notification for numerous events such as method entry and exit, exception, field access and modification, thread start and end, and/or other events. Examples of JVMTI built-in methods include GetMethodName to obtain the name of an invoked method, GetThreadInfo to obtain information for a specific thread, GetClassSignature to obtain information about the class of an object, GetStackTrace to obtain information about the stack including information about stack frames, and/or other methods. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through java.lang.Runtime class that provides an interface for application tracing or profiling. Examples of methods provided in java.lang.Runtime that can be used to obtain an application's instruction sets, data, and/or other information include tracemethodcalls, traceinstructions, and/or other methods. These methods prompt the Java Virtual Machine to output trace information for a method or instruction in the virtual machine as it is executed. The destination of trace output may be system dependent and include a file, a listener, and/or other destinations where Acquisition Interface 120, Artificial Intelligence Unit 110, and/or other disclosed elements can access needed information. In addition to tracing or profiling tools native to their respective computing systems and/or platforms, many independent tools exist that provide tracing or profiling functionalities on more than one computing system and/or platform. Examples of these tools include Pin, DynamoRIO, KernInst, DynInst, Kprobes, OpenPAT, DTrace, SystemTap, and/or others. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through logging tools of the platform and/or operating system on which an application runs. Some logging tools may include nearly full feature sets of the tracing or profiling tools previously described. In one example, Visual Basic enables logging of runtime messages through its Microsoft.VisualBasic.Logging namespace that provides a log listener where the log listener may direct logging output to a file and/or other target. In another example, Java enables logging through its java.util.logging class. In some aspects, obtaining an application's instruction sets, data, and/or other information can be implemented through logging capabilities of the operating system on which an application runs. For example, Windows NT features centralized log service that applications and operating-system components can utilize to report their events including any messages. Windows NT provides functionalities for system, application, security, and/or other logging. An application log may include events logged by applications. Windows NT, for example, may include support for defining an event source (i.e. application that created the event, etc.). Windows Vista, for example, supports a structured XML log-format and designated log types to allow applications to more precisely log events and to help interpret the events. Examples of different types of event logs include administrative, operational, analytic, debug, and/or other log types including any of their subcategories. Examples of event attributes that can be utilized include eventID, level, task, opcode, keywords, and/or other event attributes. Windows wevtutil tool enables access to events, their structures, registered event publishers, and/or their configuration even before the events are fired. Wevtutil supports capabilities such as retrieval of the names of all logs on a computing device; retrieval of configuration information for a specific log; retrieval of event publishers on a computing device; reading events from an event log, from a log file, or using a structured query; exporting events from an event log, from a log file, or using a structured query to a specific target; and/or other capabilities. Operating system logs can be utilized solely if they contain sufficient information on an application's instruction sets, data, and/or other information. Alternatively, operating system logs can be utilized in combination with another source of information (i.e. trace information, call stack, processor registers, memory, etc.) to reconstruct the application's instruction sets, data, and/or other information needed for Artificial Intelligence Unit 110 and/or other elements. In addition to logging capabilities native to their respective platforms and/or operating systems, many independent tools exist that provide logging on different platforms and/or operating systems. Examples of these tools include Log4j, Logback, SmartInspect, NLog, log4net, Microsoft Enterprise Library, ObjectGuy Framework, and/or others.

Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through tracing or profiling the operating system on which an application runs. As in tracing or profiling an application, one of the techniques that can be utilized includes adding instrumentation code to the operating system's source code. Such instrumentation code can be added to the operating system's source code before kernel compilation or recompilation, for instance. This type of instrumentation may involve defining or finding locations in the operating system's source code where instrumentation code may be inserted. Kernel instrumentation can also be performed without the need for kernel recompilation or rebooting. In some aspects, instrumentation code can be added at locations of interest through binary rewriting of compiled kernel code. In other aspects, kernel instrumentation can be performed dynamically where instrumentation code is added and/or removed where needed at runtime. For instance, dynamic instrumentation may overwrite kernel code with a branch instruction that redirects execution to instrumentation code or instrumentation routine. In yet other aspects, kernel instrumentation can be performed using just-in-time (JIT) dynamic instrumentation where execution may be redirected to a copy of kernel's code segment that includes instrumentation code. This type of instrumentation may include a JIT compiler and creation of a copy of the original code segment having instrumentation code or calls to instrumentation routines embedded into the original code segment. Instrumentation of the operating system may enable total system visibility including visibility into an application's behavior by enabling generation of low level trace information. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through tracing or profiling the processor on which an application runs. For example, some Intel processors provide Intel Processor Trace (i.e. Intel PT, etc.), a low-level tracing feature that enables recording executed instruction sets, and/or other data or information of one or more applications. Intel PT is facilitated by the Processor Trace Decoder Library along with its related tools. Intel PT is a low-overhead execution tracing feature that records information about application execution on each hardware thread using dedicated hardware facilities. The recorded execution/trace information is collected in data packets that can be buffered internally before being sent to a memory subsystem or another system or element (i.e. Acquisition Interface 120, Artificial Intelligence Unit 110, etc.). Intel PT also enables navigating the recorded execution/trace information via reverse stepping commands. Intel PT can be included in an operating system's core files and provided as a feature of the operating system. Intel PT can trace globally some or all applications running on an operating system. Acquisition Interface 120 or Artificial Intelligence Unit 110 can read or obtain the recorded execution/trace information from Intel PT. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through branch tracing or profiling. Branch tracing may include an abbreviated instruction trace in which only the successful branch instruction sets are traced or recorded. Branch tracing can be implemented through utilizing dedicated processor commands, for example. Executed branches may be saved into special branch trace store area of memory. With the availability and reference to a compiler listing of the application together with branch trace information, a full path of executed instruction sets can be reconstructed. The full path can also be reconstructed with a memory dump (containing the program storage) and branch trace information. In some aspects, branch

tracing can be utilized for pre-learning or automated learning of an application's instruction sets, data, and/or other information where a number of application simulations (i.e. simulations of likely/common operations, etc.) are performed. As such, the application's operation can be learned automatically saving the time that would be needed to learn the application's operation directed by a user. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through assembly language. Assembly language is a low-level programming language for a computer or other programmable device in which there is a strong correlation between the language and the architecture's machine instruction sets. Syntax, addressing modes, operands, and/or other elements of an assembly language instruction set may translate directly into numeric (i.e. binary, etc.) representations of that particular instruction set. Because of this direct relationship with the architecture's machine instruction sets, assembly language can be a powerful tool for tracing or profiling an application's execution in processor registers, memory, and/or other computing system components. For example, using assembly language, memory locations of a loaded application can be accessed, instrumented, and/or otherwise manipulated. In some aspects, assembly language can be used to rewrite or overwrite original in-memory instruction sets of an application with instrumentation instruction sets. In other aspects, assembly language can be used to redirect application's execution to instrumentation routine/subroutine or other code segment elsewhere in memory by inserting a jump into the application's in-memory code, by redirecting program counter, or by other techniques. Some operating systems may implement protection from changes to applications loaded into memory. Operating system, processor, or other low level commands such as Linux mprotect command or similar commands in other operating systems may be used to unprotect the protected locations in memory before the change. In yet other aspects, assembly language can be used to obtain instruction sets, data, and/or other information through accessing and/or reading instruction register, program counter, other processor registers, memory locations, and/or other components of a computing system. In yet other aspects, high-level programming languages may call or execute an external assembly language program to facilitate obtaining instruction sets, data, and/or other information as previously described. In yet other aspects, relatively low-level programming languages such as C may allow embedding assembly language directly in their source code such as, for example, using asm keyword of C. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, it may be sufficient to obtain user or other inputs, variables, parameters, and/or other data in some procedural, simple object oriented, or other applications. In one instance, a simple procedural application executes a sequence of instruction sets until the end of the program. During its execution, the application may receive user or other input, store the input in a variable, and perform calculations using the variable to reach a result. The value of the variable can be obtained or traced. In another instance, a more complex procedural application comprises one or more functions/routines/subroutines each of which may include a sequence of instruction sets. The application may execute a main sequence of instruction sets with a branch to a function/routine/subroutine. During its execution, the application may receive user or other input, store the input in a variable, and pass the variable as a parameter to the function/routine/subroutine. The function/routine/subroutine may perform calculations using the parameter and return a value that the rest of the application can use to reach a result. The value of the variable or parameter passed to the function/routine/subroutine, and/or return value can be

obtained or traced. Values of user or other inputs, variables, parameters, and/or other items of interest can be obtained through previously described tracing, instrumentation, and/or other techniques. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

Referring to Fig. 3, in yet another example, obtaining instruction sets, data, and/or other information may be implemented through tracing, profiling, or sampling of instruction sets or data in processor registers, memory, or other computing system components where instruction sets, data, and/or other information may be stored or utilized. For example, Instruction Register 212 may be part of Processor 11 and it may store the instruction set currently being executed or decoded. In some processors, Program Counter 211 (also referred to as instruction pointer, instruction address register, instruction counter, or part of instruction sequencer) may be incremented after fetching an instruction set, and it may hold or point to the memory address of the next instruction set to be executed. In a processor where the incrementation precedes the fetch, Program Counter 211 may point to the current instruction set being executed. In the instruction cycle, an instruction set may be loaded into Instruction Register 212 after Processor 11 fetches it from location in Memory 12 pointed to by Program Counter 211. Instruction Register 212 may hold the instruction set while it is decoded by Instruction Decoder 213, prepared, and executed. In some aspects, data (i.e. operands, etc.) needed for instruction set execution may be loaded from Memory 12 into a register within Register Array 214. In other aspects, the data may be loaded directly into Arithmetic Logic Unit 215. For instance, as instruction sets pass through Instruction Register 212 during application execution, they may be transmitted to Acquisition Interface 120 as shown. Examples of the steps in execution of a machine instruction set may include decoding the opcode (i.e. portion of a machine instruction set that may specify the operation to be performed), determining where the operands may be located (depending on architecture, operands may be in registers, the stack, memory, I/O ports, etc.), retrieving the operands, allocating processor resources to execute the instruction set (needed in some types of processors), performing the operation indicated by the instruction set, saving the results of execution, and/or other execution steps. Examples of the types of machine instruction sets that can be utilized include arithmetic, data handling, logical, program control, as well as special and/or other instruction set types. In addition to the ones described or shown, examples of other computing system or processor components that can be used during an instruction cycle include memory address register (MAR) that may hold the address of a memory block to be read from or written to; memory data register (MDR) that may hold data fetched from memory or data waiting to be stored in memory; data registers that may hold numeric values, characters, small bit arrays, or other data; address registers that may hold addresses used by instruction sets that indirectly access memory; general purpose registers (GPRs) that may store both data and addresses; conditional registers that may hold truth values often used to determine whether some instruction set should or should not be executed; floating point registers (FPRs) that may store floating point numbers; constant registers that may hold read-only values such as zero, one, or pi; special purpose registers (SPRs) such as status register, program counter, or stack pointer that may hold information on program state; machine-specific registers that may store data and settings related to a particular processor; Register Array 214 that may include an array of any number of processor registers; Arithmetic Logic Unit 215 that may perform arithmetic and logic operations; control unit that may direct processor’s operation; and/or other circuits or components. Tracing, profiling, or sampling of processor registers, memory, or other computing system components can be implemented in a program, combination of hardware and program, or purely hardware system. Dedicated hardware may be built to perform tracing, profiling, or sampling of processor registers

or any computing system components with marginal or no impact to computing overhead.

One of ordinary skill in art will recognize that Fig. 3 depicts one of many implementations of processor or computing system components, and that various additional components can be included, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate implementations. Processor or computing system components may be arranged or connected differently in alternate implementations. Processor or computing system components may also be connected with external elements using various connections. For instance, the connection between Instruction Register 212 and Acquisition Interface 120 may include any number or types of connections such as, for example, a dedicated connection for each bit of Instruction Register 212 (i.e. 32 connections for a 32 bit Instruction Register 212, etc.). Any of the described or other connections or interfaces may be implemented among any processor or computing system components and Acquisition Interface 120 or other elements.

Referring to Figs. 4A-4B, in yet another example, obtaining instruction sets, data, and/or other information may be implemented through tracing, profiling, or sampling of Logic Circuit 250. While Processor 11 includes any type or embodiment of logic circuit, Logic Circuit 250 is described separately here to offer additional detail on its functioning. Some Devices 98 may not need the processing capabilities of an entire Processor 11, but instead a more tailored Logic Circuit 250. Examples of such Devices 98 include home appliances, audio or video electronics, vehicle systems, toys, industrial machines, robots, and/or others. Logic Circuit 250 comprises the functionality for performing logic operations. Logic Circuit 250 comprises the functionality for performing logic operations using the circuit's inputs and producing outputs based on the logic operations performed on the inputs. Logic Circuit 250 may generally be implemented using transistors, diodes, and/or other electronic switches, but can also be constructed using vacuum tubes, electromagnetic relays (relay logic), fluidic logic, pneumatic logic, optics, molecules, or even mechanical elements. In some aspects, Logic Circuit 250 may be or include a microcontroller, field-programmable gate array (FPGA), application-specific integrated circuit (ASIC), and/or other computing circuit or device. In other aspects, Logic Circuit 250 may be or include any circuit or device comprising one or more logic gates, one or more transistors, one or more switches, and/or one or more other logic components. In further aspects, Logic Circuit 250 may be or include any integrated or other circuit or device that can perform logic operations. Logic may generally refer to Boolean logic utilized in binary operations, but other logics can also be used. Input into Logic Circuit 250 may include or refer to a value inputted into the Logic Circuit 250, therefore, these terms may be used interchangeably herein depending on context. In one example, Logic Circuit 250 may perform some logic operations using four input values and produce two output values. As the four input values are delivered to or received by Logic Circuit 250, they may be obtained by Acquisition Interface 120 through the four hardwired connections as shown in Fig. 4A. In another example, Logic Circuit 250 may perform some logic operations using four input values and produce two output values. As the two output values are generated by or transmitted out of Logic Circuit 250, they may be obtained by Acquisition Interface 120 through the two hardwired connections as shown in Fig. 4B. In a further example, instead of or in addition to obtaining input and/or output values of Logic Circuit 250, the state of Logic Circuit 250 may be obtained by reading or accessing values from one or more Logic Circuit's 250 internal components such as registers, memories, buses, and/or others (i.e. similar to the previously described tracing, profiling, and/or sampling of Processor 11 components, etc.). Tracing, profiling, or sampling of Logic Circuit 250 can be implemented in a program, combination of hardware and program, or purely hardware system. Dedicated

hardware may be built to perform tracing, profiling or sampling of Logic Circuit 250 with marginal or no impact to computing overhead. Any of the elements and/or techniques for tracing, profiling, or sampling of Logic Circuit 250 can similarly be implemented with Processor 11 and/or other processing elements. In some designs, VSADO Unit 100 may include clamps and/or other elements to attach VSADO Unit 100 to inputs (i.e. input wires, etc.) into and/or outputs (i.e. output wires, etc.) from Logic Circuit 250. Such clamps and/or attachment elements enable seamless attachment of VSADO Unit 100 to any circuit or computing device without the need to redesign or alter the circuit or computing device.

In some embodiments, VSADO Unit 100 may learn input values directly from an actuator (previously described, not shown). For example, Logic Circuit 250 or other processing element may control an actuator that enables Device 98 to perform mechanical, physical, and/or other operations. An actuator may receive one or more input values or control signals from Logic Circuit 250 or other processing element directing the actuator to perform specific operations. As one or more input values or control signals are delivered to or received by the actuator, they may be obtained by Acquisition Interface 120 as previously described with respect to obtaining input values of Logic Circuit 250. Specifically, for instance, one or more input values or control signals of an actuator may be obtained by Acquisition Interface 120 via hardwired or other connections.

One of ordinary skill in art will recognize that Figs. 4A-4B depict one of many implementations of Logic Circuit 250 and that any number of input and/or output values can be utilized in alternate implementations. One of ordinary skill in art will also recognize that Logic Circuit 250 may include any number and/or combination of logic components to implement any logic operations.

Other additional techniques or elements may be utilized as needed for obtaining instruction sets, data, and/or other information, or some of the disclosed techniques or elements may be excluded, or a combination thereof may be utilized in alternate embodiments.

Referring to Figs. 5A-5C, some embodiments of Instruction Sets 526 are illustrated. In some aspects, Instruction Set 526 includes one or more instructions or commands of Application Program 18. For example, Instruction Set 526 may include one or more instructions or commands of a high-level programming language such as Java or SQL, a low-level language such as assembly or machine language, an intermediate language or construct such as bytecode, and/or any other language or construct. In other aspects, Instruction Set 526 includes one or more inputs into and/or outputs from Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element. In further aspects, Instruction Set 526 includes one or more values or states of registers and/or other components of Logic Circuit 250, Processor 11, and/or other processing element. In general, Instruction Set 526 may include one or more instructions, commands, keywords, symbols (i.e. parentheses, brackets, commas, semicolons, etc.), operators (i.e. =, <, >, etc.), variables, values, objects, data structures, functions (i.e. Function1(), FIRST(), MIN(), SQRT(), etc.), parameters, states, signals, inputs, outputs, characters, digits, references thereto, and/or other components for performing an operation.

In an embodiment shown in Fig. 5A, Instruction Set 526 includes code of a high-level programming language (i.e. Java, C++, etc.) comprising the following function call construct: Function1 (Parameter1, Parameter2, Parameter3, ...). An example of a function call applying the above construct includes the following Instruction Set 526: moveTo(Object1, 29, 17). The function or reference thereto “moveTo(Object1, 29, 17)” may be an Instruction Set 526 directing Object1 to move to a location with coordinates 29 and 17, for example. In another embodiment

shown in Fig. 5B, Instruction Set 526 includes structured query language (SQL). In a further embodiment shown in Fig. 5C, Instruction Set 526 includes bytecode (i.e. Java bytecode, Python bytecode, CLR bytecode, etc.). In a further embodiment shown in Fig. 5D, Instruction Set 526 includes assembly code. In a further embodiment shown in Fig. 5E, Instruction Set 526 includes machine code.

Referring to Figs. 6A-6B, some embodiments of Extra Information 527 (also referred to as Extra Info 527) are illustrated. In an embodiment shown in Fig. 6A, Digital Picture 525 may include or be associated with Extra Info 527. In an embodiment shown in Fig. 6B, Instruction Set 526 may include or be associated with Extra Info 527.

Extra Info 527 comprises the functionality for storing any information useful in comparisons or decision making performed in autonomous device operation, and/or other functionalities. One or more Extra Infos 527 can be stored in, appended to, or associated with a Digital Picture 525, Instruction Set 526, and/or other element. In some embodiments, the system can obtain Extra Info 527 at a time of capturing or receiving of Digital Picture 525. In other embodiments, the system can obtain Extra Info 527 at a time of acquisition of Instruction Set 526. In general, the system or any element thereof can obtain Extra Info 527 at any time. Examples of Extra Info 527 include time information, location information, computed information, observed information, sensory information, contextual information, and/or other information. Any information can be utilized that can provide information for enhanced comparisons or decision making performed in autonomous device operation. Which information is utilized and/or stored in Extra Info 527 can be set by a user, by VSADO system administrator, or automatically by the system. Extra Info 527 may include or be referred to as contextual information, and vice versa. Therefore, these terms may be used interchangeably herein depending on context.

In some aspects, time information (i.e. time stamp, etc.) can be utilized and/or stored in Extra Info 527. Time information can be useful in comparisons or decision making performed in autonomous device operation related to a specific time period as Device 98 may be required to perform specific operations at certain parts of day, month, year, and/or other time periods. Time information can be obtained from the system clock, online clock, oscillator, or other time source. In one example, a thermostat device may be directed to turn heat on in the morning and/or turn heat off during the day. In a further example, a personal computer device may be directed to start or stop an application program or process on a particular day of the month. In general, Extra Info 527 may include time information related to when Device 98 performed an operation. In other aspects, location information (i.e. coordinates, address, etc.) can be utilized and/or stored in Extra Info 527. Location information can be useful in comparisons or decision making performed in autonomous device operation related to a specific place as Device 98 may be required to perform specific operations at certain places. Location information can be obtained from a positioning system (i.e. radio signal triangulation in smartphones or tablets, GPS capabilities, etc.) if one is available. In one example, a smartphone device may be directed to engage a vibrate mode in a school or house of worship. In another example, a vehicle may be directed to turn right at a particular road crossing. In general, Extra Info 527 may include location information related to where Device 98 performed an operation. In further aspects, computed information can be utilized and/or stored in Extra Info 527. Computed information can be useful in comparisons or decision making performed in autonomous device operation where information can be calculated, inferred, or derived from other available information. VSADO Unit 100 may include computational functionalities to create Extra Info 527 by performing calculations or inferences using other information. In one example, Device's 98 speed can be computed or estimated from the Device's 98 location and/or time information. In another example, Device's 98

bearing (i.e. angle or direction of movement, etc.) can be computed or estimated from the Device's 98 location information by utilizing Pythagorean theorem, trigonometry, and/or other theorems, formulas, or disciplines. In a further example, speeds, bearings, distances, and/or other properties of objects around Device 98 can similarly be computed or inferred, thereby providing geo-spatial and situational awareness and/or capabilities to the Device 98. In further aspects, observed information can be utilized and/or stored in Extra Info 527. Observed information can be useful in comparisons or decision making performed in autonomous device operation related to a specific object or environment as Device 98 may be required to perform certain operations around specific objects or in specific environments. For example, an object or environment can be recognized by processing one or more Digital Pictures 525 from Picture Capturing Apparatus 90. Any features, functionalities, and embodiments of Picture Recognizer 350 (later described) can be utilized for such recognizing. In one example, book shelves recognized in the background of one or more Digital Pictures 525 from Picture Capturing Apparatus 90 may indicate a library or book store. In another example, trees recognized in the background of one or more Digital Pictures 525 from Picture Capturing Apparatus 90 may indicate a park. In a further example, a pedestrian recognized in one or more Digital Pictures 525 from Picture Capturing Apparatus 90 may indicate a street. In further aspects, sensory information can be utilized and/or stored in Extra Info 527. Examples of sensory information include acoustic information, visual information, tactile information, and/or others. Sensory information can be useful in comparisons or decision making performed in autonomous device operation related to a specific object or environment as Device 98 may be required to perform certain operations around specific objects or in specific environments. For example, an object or environment can be recognized by processing digital sound from a sound capturing apparatus (i.e. microphone, etc., not shown). Any features, functionalities, and embodiments of a speech or sound recognizer (not shown) can be utilized for such recognizing. In one example, sound of waves recognized in digital sound from a sound capturing apparatus may indicate a beach. In another example, sound of a horn recognized in digital sound from a sound capturing apparatus may indicate a proximal vehicle. In some designs where acoustic information includes one or more digital sound samples of Device's 98 surrounding captured by a sound capturing apparatus, the digital sound samples can be learned and/or used similar to Digital Pictures 525 of Device's 98 visual surrounding. In such designs, both Digital Pictures 525 and digital sound samples of a device's surrounding can be learned and/or used for autonomous device operation. In further aspects, other information can be utilized and/or stored in Extra Info 527. Examples of such other information include user specific information (i.e. skill level, age, gender, etc.), group user information (i.e. access level, etc.), version of Application Program 18, the type of Application Program 18, the type of Processor 11, the type of Logic Circuit 250, the type of Device 98, and/or other information.

Referring to Fig. 7, an embodiment where VSADO Unit 100 is part of or operating on Processor 11 is illustrated. In one example, VSADO Unit 100 may be a hardware element or circuit embedded or built into Processor 11. In another example, VSADO Unit 100 may be a program operating on Processor 11.

Referring to Fig. 8, an embodiment where VSADO Unit 100 resides on Server 96 accessible over Network 95 is illustrated. Any number of Devices 98 may connect to such remote VSADO Unit 100 and the remote VSADO Unit 100 may learn their operations in various visual surroundings. In turn, any number of Devices 98 can utilize the remote VSADO Unit 100 for autonomous operation. A remote VSADO Unit 100 can be offered as a network service (i.e. online application, etc.). In some aspects, a remote VSADO Unit 100 (i.e. global VSADO Unit 100, etc.) may reside on the Internet and be available to all the world's Devices 98 configured to transmit their operations in various

visual surroundings and/or configured to utilize the remote VSADO Unit 100 for autonomous operation. Server 96 may be or include any type or form of a remote computing device such as an application server, a network service server, a cloud server, a cloud, and/or other remote computing device. Server 96 may include any features, functionalities, and embodiments of the previously described Computing Device 70. It should be understood that Server 96 does not have to be a separate computing device and that Server 96, its elements, or its functionalities can be implemented on Device 98. Network 95 may include various networks, connection types, protocols, interfaces, APIs, and/or other elements or techniques known in art all of which are within the scope of this disclosure. Any of the previously described networks, network or connection types, networking interfaces, and/or other networking elements or techniques can similarly be utilized. Any of the disclosed elements may reside on Server 96 in alternate implementations. In one example, Artificial Intelligence Unit 110 can reside on Server 96 and Acquisition Interface 120 and/or Modification Interface 130 can reside on Device 98. In another example, Knowledgebase 530 can reside on Server 96 and the rest of the elements of VSADO Unit 100 can reside on Device 98. Any other combination of local and remote elements can be implemented.

Referring to Fig. 9, an embodiment where Picture Capturing Apparatus 90 is part of Remote Device 97 accessible over Network 95 is illustrated. In such embodiments, VSADO Unit 100 may learn Device's 98 operation based on another device's visual surrounding. Such embodiments can be utilized, for instance, in any situation where one device controls (i.e. remote control, etc.) another device, any situation where some or all of the processing is on one device and picture capturing capabilities are on another device, and/or other situations. In one example, a drone controlling device (i.e. Device 98) may receive its visual input from a camera on the drone (i.e. Remote Device 97). In another example, a toy controlling device (i.e. Device 98) may receive its visual input from a camera on the toy (i.e. Remote Device 97). In a further example, a people or crowd analyzing computing device (i.e. Device 98) may receive its visual input from a camera of a monitoring device (i.e. Remote Device 97). Any of the disclosed elements in addition to Picture Capturing Apparatus 90 may reside on Remote Device 97 in alternate implementations as previously described with respect to Server 96.

Referring to Fig. 10, an embodiment of VSADO Unit 100 comprising Picture Recognizer 350 is illustrated. VSADO Unit 100 can utilize Picture Recognizer 350 to detect or recognize persons, objects, and/or their activities in one or more digital pictures from Picture Capturing Apparatus 90. In general, VSADO Unit 100 and/or other disclosed elements can use Picture Recognizer 350 for any operation supported by Picture Recognizer 350. Picture Recognizer 350 comprises the functionality for detecting or recognizing persons or objects in visual data. Picture Recognizer 350 comprises the functionality for detecting or recognizing activities in visual data. Picture Recognizer 350 comprises the functionality for tracking persons, objects, and/or their activities in visual data. Picture Recognizer 350 comprises other disclosed functionalities. Visual data includes digital motion pictures, digital still pictures (i.e. bitmaps, etc.), and/or other visual data. Examples of file formats that can be utilized to store visual data include AVI, DivX, MPEG, JPEG, GIF, TIFF, PNG, PDF, and/or other file formats. Picture Recognizer 350 may detect or recognize a person and/or his/her activities as well as track the person and/or his/her activities in one or more digital pictures or streams of digital pictures (i.e. motion pictures, video, etc.). Picture Recognizer 350 may detect or recognize a human head or face, upper body, full body, or portions/combinations thereof. In some aspects, Picture Recognizer 350 may detect or recognize persons, objects, and/or their activities from a digital picture by comparing regions of pixels from the digital picture with collections of pixels comprising known persons, objects, and/or their

activities. The collections of pixels comprising known persons, objects, and/or their activities can be learned or manually, programmatically, or otherwise defined. The collections of pixels comprising known persons, objects, and/or their activities can be stored in any data structure or repository (i.e. one or more files, database, etc.) that resides locally on Device 98, or remotely on a remote computing device (i.e. server, cloud, etc.) accessible over a network. In other aspects, Picture Recognizer 350 may detect or recognize persons, objects, and/or their activities from a digital picture by comparing features (i.e. lines, edges, ridges, corners, blobs, regions, etc.) of the digital picture with features of known persons, objects, and/or their activities. The features of known persons, objects, and/or their activities can be learned or manually, programmatically, or otherwise defined. The features of known persons, objects, and/or their activities can be stored in any data structure or repository (i.e. neural network, one or more files, database, etc.) that resides locally on Device 98, or remotely on a remote computing device (i.e. server, cloud, etc.) accessible over a network. Typical steps or elements in a feature oriented picture recognition include pre-processing, feature extraction, detection/segmentation, decision-making, and/or others, or a combination thereof, each of which may include its own sub-steps or sub-elements depending on the application. In further aspects, Picture Recognizer 350 may detect or recognize multiple persons, objects, and/or their activities from a digital picture using the aforementioned pixel or feature comparisons, and/or other detection or recognition techniques. For example, a picture may depict two persons in two of its regions both of whom Picture Recognizer 350 can detect simultaneously. In further aspects, where persons, objects, and/or their activities span multiple pictures, Picture Recognizer 350 may detect or recognize persons, objects, and/or their activities by applying the aforementioned pixel or feature comparisons and/or other detection or recognition techniques over a stream of digital pictures (i.e. motion picture, video, etc.). For example, once a person is detected in a digital picture (i.e. frame, etc.) of a stream of digital pictures (i.e. motion picture, video, etc.), the region of pixels comprising the detected person or the person's features can be searched in other pictures of the stream of digital pictures, thereby tracking the person through the stream of digital pictures. In further aspects, Picture Recognizer 350 may detect or recognize a person's activities by identifying and/or analyzing differences between a detected region of pixels of one picture (i.e. frame, etc.) and detected regions of pixels of other pictures in a stream of digital pictures. For example, a region of pixels comprising a person's face can be detected in multiple consecutive pictures of a stream of digital pictures (i.e. motion picture, video, etc.). Differences among the detected regions of the consecutive pictures may be identified in the mouth part of the person's face to indicate smiling or speaking activity. Any technique for recognizing speech from mouth/lip movements can be used in this and other examples. In further aspects, Picture Recognizer 350 may detect or recognize persons, objects, and/or their activities using one or more artificial neural networks, which may include statistical techniques. Examples of artificial neural networks that can be used in Picture Recognizer 350 include convolutional neural networks (CNNs), time delay neural networks (TDNNs), deep neural networks, and/or others. In one example, picture recognition techniques and/or tools involving convolutional neural networks may include identifying and/or analyzing tiled and/or overlapping regions or features of a digital picture, which may then be used to search for pictures with matching regions or features. In another example, features of different convolutional neural networks responsible for spatial and temporal streams can be fused to detect persons, objects, and/or their activities in streams of digital pictures (i.e. motion pictures, videos, etc.). In general, Picture Recognizer 350 may include any machine learning, deep learning, and/or other artificial intelligence techniques. Any other techniques known in art can be utilized in Picture Recognizer 350. For example, thresholds for similarity,

statistical, and/or optimization techniques can be utilized to determine a match in any of the above-described detection or recognition techniques. Picture Recognizer 350 comprises any features, functionalities, and embodiments of Similarity Comparison 125 (later described).

In some exemplary embodiments, object recognition techniques and/or tools such as OpenCV (Open Source Computer Vision) library, CamFind API, Kooaba, 6px API, Dextro API, and/or others can be utilized for detecting or recognizing objects (i.e. objects, animals, people, etc.) in digital pictures. In some aspects, object recognition techniques and/or tools involve identifying and/or analyzing object features such as lines, edges, ridges, corners, blobs, regions, and/or their relative positions, sizes, shapes, etc., which may then be used to search for pictures with matching features. For example, OpenCV library can detect an object (i.e. car, pedestrian, door, building, animal, person, etc.) in one or more digital pictures captured by Picture Capturing Apparatus 90 or stored in an electronic repository, which can then be utilized in VSADO Unit 100, Artificial Intelligence Unit 110, and/or other elements.

In other exemplary embodiments, facial recognition techniques and/or tools such as OpenCV (Open Source Computer Vision) library, Animetrics FaceR API, Lambda Labs Facial Recognition API, Face++ SDK, Neven Vision (also known as N-Vision) Engine, and/or others can be utilized for detecting or recognizing faces in digital pictures. In some aspects, facial recognition techniques and/or tools involve identifying and/or analyzing facial features such as the relative position, size, and/or shape of the eyes, nose, cheekbones, jaw, etc., which may then be used to search for pictures with matching features. For example, FaceR API can detect a person's face in one or more digital pictures captured by Picture Capturing Apparatus 90 or stored in an electronic repository, which can then be utilized in VSADO Unit 100, Artificial Intelligence Unit 110, and/or other elements.

Referring to Fig. 11, an embodiment of Artificial Intelligence Unit 110 is illustrated. Artificial Intelligence Unit 110 comprises interconnected Knowledge Structuring Unit 520, Knowledgebase 530, Decision-making Unit 540, and Confirmation Unit 550. Other additional elements can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate embodiments.

Artificial Intelligence Unit 110 comprises the functionality for learning Device's 98 operation in various visual surroundings. Artificial Intelligence Unit 110 comprises the functionality for learning one or more digital pictures correlated with any instruction sets, data, and/or other information. In some aspects, Artificial Intelligence Unit 110 comprises the functionality for learning one or more Digital Pictures 525 of Device's 98 surrounding correlated with any Instruction Sets 526 and/or Extra Info 527. In other aspects, Artificial Intelligence Unit 110 comprises the functionality for learning one or more Digital Pictures 525 of Device's 98 surrounding some of which may not be correlated with any Instruction Sets 526 and/or Extra Info 527. Further, Artificial Intelligence Unit 110 comprises the functionality for anticipating Device's 98 operation in various visual surroundings. Artificial Intelligence Unit 110 comprises the functionality for anticipating one or more instruction sets, data, and/or other information. Artificial Intelligence Unit 110 comprises the functionality for anticipating one or more Instruction Sets 526 based on one or more incoming Digital Pictures 525 of Device's 98 surrounding. Artificial Intelligence Unit 110 comprises the functionality for anticipating one or more Instruction Sets 526 to be used or executed in Device's 98 autonomous operation. Artificial Intelligence Unit 110 also comprises other disclosed functionalities.

Knowledge Structuring Unit 520, Knowledgebase 530, and Decision-making Unit 540 are described later.

Confirmation Unit 550 comprises the functionality for confirming, modifying, evaluating (i.e. rating, etc.), and/or canceling one or more anticipatory Instruction Sets 526, and/or other functionalities. Confirmation Unit 550 is an optional element that can be omitted depending on implementation. In some embodiments, Confirmation Unit 550 can serve as a means of confirming anticipatory Instruction Sets 526. For example, Decision-making Unit 540 may determine one or more anticipatory Instruction Sets 526 and provide them to User 50 for confirmation. User 50 may be provided with an interface (i.e. graphical user interface, selectable list of anticipatory Instruction Sets 526, etc.) to approve or confirm execution of the anticipatory Instruction Sets 526. In some aspects, Confirmation Unit 550 can automate User 50 confirmation. In one example, if one or more incoming Digital Pictures 525 from Picture Capturing Apparatus 90 and one or more Digital Pictures 525 from a Knowledge Cell 800 were found to be a perfect or highly similar match, anticipatory Instruction Sets 526 correlated with the one or more Digital Pictures 525 from the Knowledge Cell 800 can be automatically executed without User's 50 confirmation. Conversely, if one or more incoming Digital Pictures 525 from Picture Capturing Apparatus 90 and one or more Digital Pictures 525 from a Knowledge Cell 800 were found to be less than a highly similar match, anticipatory Instruction Sets 526 correlated with the one or more Digital Pictures 525 from the Knowledge Cell 800 can be presented to User 50 for confirmation and/or modifying. A threshold that defines a highly or otherwise similar match can be utilized in such implementations. Such threshold can be defined by a user, by VSADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. In other embodiments, Confirmation Unit 550 can serve as a means of modifying or editing anticipatory Instruction Sets 526. For example, Decision-making Unit 540 may determine one or more anticipatory Instruction Sets 526 and provide them to User 50 for modification. User 50 may be provided with an interface (i.e. graphical user interface, etc.) to modify the anticipatory Instruction Sets 526 before their execution. In further embodiments, Confirmation Unit 550 can serve as a means of evaluating or rating anticipatory Instruction Sets 526 if they matched User's 50 intended operation. For example, Decision-making Unit 540 may determine one or more anticipatory Instruction Sets 526, which the system may automatically execute. User 50 may be provided with an interface (i.e. graphical user interface, etc.) to rate (i.e. on a scale from 0 to 1, etc.) how well Decision-making Unit 540 predicted the executed anticipatory Instruction Sets 526. In some aspects, rating can be automatic and based on a particular function or method that rates how well the anticipatory Instruction Sets 526 matched the desired operation. In one example, a rating function or method can assign a higher rating to anticipatory Instruction Sets 526 that were least modified in the confirmation process. In another example, a rating function or method can assign a higher rating to anticipatory Instruction Sets 526 that were canceled least number of times by User 50. Any other automatic rating function or method can be utilized. In yet other embodiments, Confirmation Unit 550 can serve as a means of canceling anticipatory Instruction Sets 526 if they did not match User's 50 intended operation. For example, Decision-making Unit 540 may determine one or more anticipatory Instruction Sets 526, which the system may automatically execute. The system may save the state of Device 98, Processor 11 (save its register values, etc.), Logic Circuit 250, Application Program 18 (i.e. save its variables, data structures, objects, location of its current instruction, etc.), and/or other processing elements before executing anticipatory Instruction Sets 526. User 50 may be provided with an interface (i.e. graphical user interface, selectable list of prior executed anticipatory Instruction Sets 526, etc.) to cancel one or more of the prior executed anticipatory Instruction Sets 526, and restore Device 98, Processor 11, Logic Circuit 250, Application Program 18, and/or other processing elements to a prior state. In some aspects,

Confirmation Unit 550 can optionally be disabled or omitted in order to provide an uninterrupted operation of Device 98, Processor 11, Logic Circuit 250, and/or Application Program 18. For example, a microwave oven may be suitable for implementing the user confirmation step, whereas, a robot or vehicle may be less suitable for implementing such interrupting step due to the real time nature of robot or vehicle operation.

Referring to Fig. 12, an embodiment of Knowledge Structuring Unit 520 correlating individual Digital Pictures 525 with any Instruction Sets 526 and/or Extra Info 527 is illustrated. Knowledge Structuring Unit 520 comprises the functionality for structuring the knowledge of a device's operation in various visual surroundings, and/or other functionalities. Knowledge Structuring Unit 520 comprises the functionality for correlating one or more Digital Pictures 525 of Device's 98 surrounding with any Instruction Sets 526 and/or Extra Info 527. Knowledge Structuring Unit 520 comprises the functionality for creating or generating Knowledge Cell 800 and storing one or more Digital Pictures 525 correlated with any Instruction Sets 526 and/or Extra Info 527 into the Knowledge Cell 800. As such, Knowledge Cell 800 comprises the functionality for storing one or more Digital Pictures 525 correlated with any Instruction Sets 526 and/or Extra Info 527. Knowledge Cell 800 includes a unit of knowledge of how Device 98 operated in a visual surrounding. Once created or generated, Knowledge Cells 800 can be used in/as neurons, nodes, vertices, or other elements in any of the data structures or arrangements (i.e. neural networks, graphs, sequences, etc.) used for storing the knowledge of Device's 98 operation in various visual surroundings, thereby facilitating learning functionalities herein. It should be noted that Extra Info 527 may be optionally used in some implementations to enable enhanced comparisons or decision making in autonomous device operation where applicable, and that Extra Info 527 can be omitted in alternate implementations.

In some embodiments, Knowledge Structuring Unit 520 receives one or more Digital Pictures 525 from Picture Capturing Apparatus 90. Knowledge Structuring Unit 520 may also receive one or more Instruction Sets 526 from Acquisition Interface 120. Knowledge Structuring Unit 520 may further receive any Extra Info 527. Although, Extra Info 527 is not shown in this and/or other figures for clarity of illustration, it should be noted that any Digital Picture 525, Instruction Set 526, and/or other element may include or be associated with Extra Info 527. Knowledge Structuring Unit 520 may correlate one or more Digital Pictures 525 with any Instruction Sets 526 and/or Extra Info 527. Knowledge Structuring Unit 520 may then create Knowledge Cell 800 and store the one or more Digital Pictures 525 correlated with Instruction Sets 526 and/or Extra Info 527 into the Knowledge Cell 800. Knowledge Cell 800 may include any data structure or arrangement that can facilitate such storing. For example, Knowledge Structuring Unit 520 may create Knowledge Cell 800ax and structure within it Digital Picture 525a1 correlated with Instruction Sets 526a1-526a3 and/or any Extra Info 527 (not shown). Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax a Digital Picture 525a2 correlated with Instruction Set 526a4 and/or any Extra Info 527 (not shown). Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax a Digital Picture 525a3 without a correlated Instruction Set 526 and/or Extra Info 527. Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax a Digital Picture 525a4 correlated with Instruction Sets 526a5-526a6 and/or any Extra Info 527 (not shown). Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax a Digital Picture 525a5 without a correlated Instruction Set 526 and/or Extra Info 527. Knowledge Structuring Unit 520 may structure within Knowledge Cell 800ax additional Digital Pictures 525 correlated with any number (including zero [i.e. uncorrelated]) of Instruction Sets 526 and/or Extra Info 527 by following the same logic as described above.

In some embodiments, Knowledge Structuring Unit 520 may correlate a Digital Picture 525 with one or more temporally corresponding Instruction Sets 526 and/or Extra Info 527. This way, Knowledge Structuring Unit 520 can structure the knowledge of Device's 98 operation at or around the time of the capturing of Digital Pictures 525 of Device's 98 surrounding. Such functionality enables spontaneous or seamless learning of Device's 98 operation in various visual surroundings as user operates the device in real life situations. In some designs, Knowledge Structuring Unit 520 may receive a stream of Instruction Sets 526 used or executed to effect Device's 98 operations as well as a stream of Digital Pictures 525 of Device's 98 surrounding as the operations are performed. Knowledge Structuring Unit 520 can then correlate Digital Pictures 525 from the stream of Digital Pictures 525 with temporally corresponding Instruction Sets 526 from the stream of Instruction Sets 526 and/or any Extra Info 527. Digital Pictures 525 without a temporally corresponding Instruction Set 526 may be uncorrelated, for instance. In some aspects, Instruction Sets 526 and/or Extra Info 527 that temporally correspond to a Digital Picture 525 may include Instruction Sets 526 used and/or Extra Info 527 obtained at the time of capturing the Digital Picture 525. In other aspects, Instruction Sets 526 and/or Extra Info 527 that temporally correspond to a Digital Picture 525 may include Instruction Sets 526 used and/or Extra Info 527 obtained within a certain time period before and/or after capturing the Digital Picture 525. For example, Instruction Sets 526 and/or Extra Info 527 that temporally correspond to a Digital Picture 525 may include Instruction Sets 526 used and/or Extra Info 527 obtained within 50 milliseconds, 1 second, 3 seconds, 20 seconds, 1 minute, 41 minutes, 2 hours, or any other time period before and/or after capturing the Digital Picture 525. Such time periods can be defined by a user, by VSADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. In other aspects, Instruction Sets 526 and/or Extra Info 527 that temporally correspond to a Digital Picture 525 may include Instruction Sets 526 used and/or Extra Info 527 obtained from the time of capturing of the Digital Picture 525 to the time of capturing of a next Digital Picture 525. In further aspects, Instruction Sets 526 and/or Extra Info 527 that temporally correspond to a Digital Picture 525 may include Instruction Sets 526 used and/or Extra Info 527 obtained from the time of capturing of a previous Digital Picture 525 to the time of capturing of the Digital Picture 525. Any other temporal relationship or correspondence between Digital Pictures 525 and correlated Instruction Sets 526 and/or Extra Info 527 can be implemented.

In some embodiments, Knowledge Structuring Unit 520 can structure the knowledge of Device's 98 operation in a visual surrounding into any number of Knowledge Cells 800. In some aspects, Knowledge Structuring Unit 520 can structure into a Knowledge Cell 800 a single Digital Picture 525 correlated with any Instruction Sets 526 and/or Extra Info 527. In other aspects, Knowledge Structuring Unit 520 can structure into a Knowledge Cell 800 any number (i.e. 2, 3, 5, 8, 19, 33, 99, 1715, 21822, 393477, 6122805, etc.) of Digital Pictures 525 correlated with any Instruction Sets 526 and/or Extra Info 527. In a special case, Knowledge Structuring Unit 520 can structure all Digital Pictures 525 correlated with any Instruction Sets 526 and/or Extra Info 527 into a single long Knowledge Cell 800. In further aspects, Knowledge Structuring Unit 520 can structure Digital Pictures 525 correlated with any Instruction Sets 526 and/or Extra Info 527 into a plurality of Knowledge Cells 800. In a special case, Knowledge Structuring Unit 520 can store periodic streams of Digital Pictures 525 correlated with any Instruction Sets 526 and/or Extra Info 527 into a plurality of Knowledge Cells 800 such as hourly, daily, weekly, monthly, yearly, or other periodic Knowledge Cells 800.

In some embodiments, Knowledge Structuring Unit 520 may be responsive to a triggering object, action, event, time, and/or other stimulus. In some aspects, the system can detect or recognize an object in Device's 98 visual surrounding, and Knowledge Structuring Unit 520 can structure into a Knowledge Cell 800 one or more Digital Pictures 525 correlated with any Instruction Sets 526 and/or Extra Info 527 related to the object. For example, Knowledge Structuring Unit 520 can structure into a Knowledge Cell 800 one or more Digital Pictures 525 of a pizza from a microwave oven (i.e. Device 98, etc.) correlated with any Instruction Sets 526 (i.e. inputs, outputs, or states of the microwave oven's microcontroller, etc.) causing the microwave oven to bake the pizza. Knowledge Structuring Unit 520 can also structure into the Knowledge Cell 800 any Extra Info 527 (i.e. time, location, computed, observed, sensory, and/or other information, etc.). In other aspects, the system can detect or recognize a specific action or operation performed by Device 98, and Knowledge Structuring Unit 520 can structure into a Knowledge Cell 800 one or more Digital Pictures 525 correlated with any Instruction Sets 526 and/or Extra Info 527 related to the action or operation. For example, Knowledge Structuring Unit 520 can structure into a Knowledge Cell 800 one or more Digital Pictures 525 depicting screwing of a screw by a robotic arm (i.e. Device 98, etc.) correlated with any Instruction Sets 526 causing the robotic arm to screw the screw. Knowledge Structuring Unit 520 can also structure into the Knowledge Cell 800 any Extra Info 527 (i.e. time, location, computed, observed, sensory, and/or other information, etc.). In further aspects, the system can detect a person in Device's 98 visual surrounding, and Knowledge Structuring Unit 520 can structure into a Knowledge Cell 800 one or more Digital Pictures 525 correlated with any Instruction Sets 526 and/or Extra Info 527 related to the person. For example, Knowledge Structuring Unit 520 can structure into a Knowledge Cell 800 one or more Digital Pictures 525 of a pedestrian in front of a vehicle (i.e. Device 98, etc.) correlated with any Instruction Sets 526 causing the vehicle to stop. Knowledge Structuring Unit 520 can also structure into the Knowledge Cell 800 any Extra Info 527 (i.e. time, location, computed, observed, sensory, and/or other information, etc.). In further aspects, the system can detect or recognize a significant change in Device's 98 visual surrounding, and Knowledge Structuring Unit 520 can structure into a Knowledge Cell 800 one or more Digital Pictures 525 correlated with any Instruction Sets 526 and/or Extra Info 527 related to the change in visual surrounding. For example, the system can detect a vehicle's (i.e. Device 98, etc.) changing direction (i.e. turning left, right, etc.) and Knowledge Structuring Unit 520 can structure into a Knowledge Cell 800 one or more Digital Pictures 525 correlated with any Instruction Sets 526 causing the change of direction. Knowledge Structuring Unit 520 can also structure into the Knowledge Cell 800 any Extra Info 527 (i.e. time, location, computed, observed, sensory, and/or other information, etc.). A vehicle's changing direction may be detected as a significant change in the vehicle's visual surrounding as the view of the vehicle's scenery changes significantly. Any features, functionalities, and embodiments of Picture Recognizer 350 can be utilized in the aforementioned detecting or recognizing. In general, Knowledge Structuring Unit 520 can structure into a Knowledge Cell 800 any Digital Pictures 525 correlated with any Instruction Sets 526 and/or Extra Info 527 related to any triggering object, action, event, time, and/or other stimulus.

In some embodiments, Device 98 may include a plurality of Picture Capturing Apparatuses 90. In one example, different Picture Capturing Apparatuses 90 may capture Digital Pictures 525 of different angles or sides of Device 98. In another example, different Picture Capturing Apparatuses 90 may be placed on different sub-devices, sub-systems, or elements of Device 98. Using multiple Picture Capturing Apparatuses 90 may provide additional visual detail in learning and/or using Device's 98 surrounding for autonomous Device 98 operation. In some designs

where multiple Picture Capturing Apparatuses 90 are utilized, multiple VSADO Units 100 can also be utilized (i.e. one VSADO Unit 100 for each Picture Capturing Apparatus 90, etc.). Digital Pictures 525 of Device's 98 surrounding can be correlated with any Instruction Sets 526 and/or Extra Info 527 as previously described. In other designs where multiple Picture Capturing Apparatuses 90 are utilized, collective Digital Pictures 525 of Device's 98 surrounding from multiple Picture Capturing Apparatuses 90 can be correlated with any Instruction Sets 526 and/or Extra Info 527.

In some embodiments, Device 98 may include a plurality of Logic Circuits 250, Processors 11, Application Programs 18, and/or other processing elements. For example, each processing element may control a sub-device, sub-system, or an element of Device 98. Using multiple processing elements may provide enhanced control over Device's 98 operation. In some designs where multiple processing elements are utilized, multiple VSADO Units 100 can also be utilized (i.e. one VSADO Unit 100 for each processing element, etc.). Digital Pictures 525 of Device's 98 surrounding can be correlated with any Instruction Sets 526 and/or Extra Info 527 as previously described. In other designs where multiple processing elements are utilized, Digital Pictures 525 of Device's 98 surrounding can be correlated with any collective Instruction Sets 526 and/or Extra Info 527 used or executed by a plurality of processing elements.

Any combination of the aforementioned multiple Picture Capturing Apparatuses 90, multiple processing elements, and/or other elements can be implemented in alternate embodiments.

Referring to Fig. 13, another embodiment of Knowledge Structuring Unit 520 correlating individual Digital Pictures 525 with any Instruction Sets 526 and/or Extra Info 527 is illustrated. In such embodiments, Knowledge Structuring Unit 520 may generate Knowledge Cells 800 each comprising a single Digital Picture 525 correlated with any Instruction Sets 526 and/or Extra Info 527.

Referring to Fig. 14, an embodiment of Knowledge Structuring Unit 520 correlating streams of Digital Pictures 525 with any Instruction Sets 526 and/or Extra Info 527 is illustrated. In some aspects, a stream of Digital Pictures 525 may include a collection, a group, a sequence, or other plurality of Digital Pictures 525. In other aspects, a stream of Digital Pictures 525 may include one or more Digital Pictures 525. In further aspects, a stream of Digital Pictures 525 may include a digital motion picture (i.e. digital video, etc.) or portion thereof. For example, Knowledge Structuring Unit 520 may create Knowledge Cell 800ax and structure within it a stream of Digital Pictures 525a1-525an correlated with Instruction Set 526a1 and/or any Extra Info 527 (not shown). Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax a stream of Digital Pictures 525b1-525bn correlated with Instruction Sets 526a2-526a4 and/or and Extra Info 527 (not shown). Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax a stream of Digital Pictures 525c1-525cn without correlated Instruction Sets 526 and/or Extra Info 527. Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax a stream of Digital Pictures 525d1-525dn correlated with Instruction Sets 526a5-526a6 and/or any Extra Info 527 (not shown). Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax additional streams of Digital Pictures 525 correlated with any number (including zero [i.e. uncorrelated]) of Instruction Sets 526 and/or Extra Info 527 by following the same logic as described above. The number of Digital Pictures 525 in some or all streams of Digital Pictures 525a1-525an, 525b1-525bn, etc. may be equal or different. It should be noted that n or other such letters or indicia may follow the sequence and/or context where they are indicated. Also, a same letter or indicia such as n may represent a different number in different elements of a drawing.

Referring to Fig. 15, another embodiment of Knowledge Structuring Unit 520 correlating streams of Digital Pictures 525 with any Instruction Sets 526 and/or Extra Info 527 is illustrated. In such embodiments, Knowledge Structuring Unit 520 may generate Knowledge Cells 800 each comprising a single stream of Digital Pictures 525 correlated with any Instruction Sets 526 and/or Extra Info 527.

Knowledgebase 530 comprises the functionality for storing the knowledge of a device's operation in various visual surroundings, and/or other functionalities. Knowledgebase 530 comprises the functionality for storing one or more Digital Pictures 525 of Device's 98 surrounding correlated with any Instruction Sets 526 and/or Extra Info 527. Knowledgebase 530 comprises the functionality for storing one or more Knowledge Cells 800 each including one or more Digital Pictures 525 of Device's 98 surrounding correlated with any Instruction Sets 526 and/or Extra Info 527. In some aspects, Digital Pictures 525 correlated with Instruction Sets 526 and/or Extra Info 527 can be stored directly within Knowledgebase 530 without using Knowledge Cells 800 as the intermediary data structures. In some embodiments, Knowledgebase 530 may be or include Neural Network 530a (later described). In other embodiments, Knowledgebase 530 may be or include Graph 530b (later described). In further embodiments, Knowledgebase 530 may be or include Collection of Sequences 530c (later described). In further embodiments, Knowledgebase 530 may be or include Sequence 533 (later described). In further embodiments, Knowledgebase 530 may be or include Collection of Knowledge Cells 530d (later described). In general, Knowledgebase 530 may be or include any data structure or arrangement capable of storing the knowledge of a device's operation in various visual surroundings. Knowledgebase 530 may reside locally on Device 98, or remotely (i.e. remote Knowledgebase 530, etc.) on a remote computing device (i.e. server, cloud, etc.) accessible over a network.

Knowledgebase 530 from one Device 98 or VSADO Unit 100 can be transferred to one or more other Devices 98 or VSADO Units 100. Therefore, the knowledge of Device's 98 operation in various visual surroundings learned on one Device 98 or VSADO Unit 100 can be transferred to one or more other Devices 98 or VSADO Units 100. In one example, Knowledgebase 530 can be copied or downloaded to a file or other repository from one Device 98 or VSADO Unit 100 and loaded or inserted into another Device 98 or VSADO Unit 100. In another example, Knowledgebase 530 from one Device 98 or VSADO Unit 100 can be available on a server accessible by other Devices 98 or VSADO Units 100 over a network. Once loaded into or accessed by a receiving Device 98 or VSADO Unit 100, the receiving Device 98 or VSADO Unit 100 can then implement the knowledge of Device's 98 operation in various visual surroundings learned on the originating Device 98 or VSADO Unit 100. This functionality enables User 50 such as a professional Device 98 operator to record his/her knowledge, methodology, or style of operating Device 98 in various visual surroundings and/or sell his/her knowledge to other users.

Referring to Fig. 16, the disclosed artificially intelligent systems, devices, and methods for learning and/or using visual surrounding for autonomous device operation may include various artificial intelligence models and/or techniques. The disclosed systems, devices, and methods are independent of the artificial intelligence model and/or technique used and any model and/or technique can be utilized to facilitate the functionalities described herein. Examples of these models and/or techniques include deep learning, supervised learning, unsupervised learning, neural networks (i.e. convolutional neural network, recurrent neural network, deep neural network, etc.), search-based, logic and/or fuzzy logic-based, optimization-based, tree/graph/other data structure-based, hierarchical, symbolic and/or sub-symbolic, evolutionary, genetic, multi-agent, deterministic, probabilistic, statistical, and/or other models and/or techniques.

In one example shown in Model A, the disclosed artificially intelligent systems, devices, and methods for learning and/or using visual surrounding for autonomous device operation may include a neural network (also referred to as artificial neural network, etc.). As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include a network of Nodes 852 (also referred to as neurons, etc.) and Connections 853 similar to that of a brain. Node 852 can store any data, object, data structure, and/or other item, or reference thereto. Node 852 may also include a function for transforming or manipulating any data, object, data structure, and/or other item. Examples of such transformation functions include mathematical functions (i.e. addition, subtraction, multiplication, division, sin, cos, log, derivative, integral, etc.), object manipulation functions (i.e. creating an object, modifying an object, deleting an object, appending objects, etc.), data structure manipulation functions (i.e. creating a data structure, modifying a data structure, deleting a data structure, creating a data field, modifying a data field, deleting a data field, etc.), and/or other transformation functions. Connection 853 may include or be associated with a value such as a symbolic label or numeric attribute (i.e. weight, cost, capacity, length, etc.). A computational model can be utilized to compute values from inputs based on a pre-programmed or learned function or method. For example, a neural network may include one or more input neurons that can be activated by inputs. Activations of these neurons can then be passed on, weighted, and transformed by a function to other neurons. Neural networks may range from those with only one layer of single direction logic to multi-layer of multi-directional feedback loops. A neural network can use weights to change the parameters of the network's throughput. A neural network can learn by input from its environment or from self-teaching using written-in rules. A neural network can be utilized as a predictive modeling approach in machine learning. An exemplary embodiment of a neural network (i.e. Neural Network 530a, etc.) is described later.

In another example shown in Model B, the disclosed artificially intelligent systems, devices, and methods for learning and/or using visual surrounding for autonomous device operation may include a graph or graph-like data structure. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include Nodes 852 (also referred to as vertices or points, etc.) and Connections 853 (also referred to as edges, arrows, lines, arcs, etc.) organized as a graph. In general, any Node 852 in a graph can be connected to any other Node 852. A Connection 853 may include unordered pair of Nodes 852 in an undirected graph or ordered pair of Nodes 852 in a directed graph. Nodes 852 can be part of the graph structure or external entities represented by indices or references. A graph can be utilized as a predictive modeling approach in machine learning. Nodes 852, Connections 853, and/or other elements or operations of a graph may include any features, functionalities, and embodiments of the aforementioned Nodes 852, Connections 853, and/or other elements or operations of a neural network, and vice versa. An exemplary embodiment of a graph (i.e. Graph 530b, etc.) is described later.

In a further example shown in Model C, the disclosed artificially intelligent systems, devices, and methods for learning and/or using visual surrounding for autonomous device operation may include a tree or tree-like data structure. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include Nodes 852 and Connections 853 (also referred to as references, edges, etc.) organized as a tree. In general, a Node 852 in a tree can be connected to any number (i.e. including zero, etc.) of children Nodes 852. A tree can be utilized as a predictive modeling approach in machine learning. Nodes 852, Connections 853, and/or other elements or operations of a tree may include any features,

functionalities, and embodiments of the aforementioned Nodes 852, Connections 853, and/or other elements or operations of a neural network and/or graph, and vice versa.

In a further example shown in Model D, the disclosed artificially intelligent systems, devices, and methods for learning and/or using visual surrounding for autonomous device operation may include a sequence or sequence-like data structure. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include a structure of Nodes 852 and/or Connections 853 organized as a sequence. In some aspects, Connections 853 may be optionally omitted from a sequence as the sequential order of Nodes 852 in a sequence may be implied in the structure. A sequence can be utilized as a predictive modeling approach in machine learning. Nodes 852, Connections 853, and/or other elements or operations of a sequence may include any features, functionalities, and embodiments of the aforementioned Nodes 852, Connections 853, and/or other elements or operations of a neural network, graph, and/or tree, and vice versa. An exemplary embodiment of a sequence (i.e. Collection of Sequences 530c, Sequence 533, etc.) is described later.

In yet another example, the disclosed artificially intelligent systems, devices, and methods for learning and/or using visual surrounding for autonomous device operation may include a search-based model and/or technique. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include searching through a collection of possible solutions. For example, a search method can search through a neural network, graph, tree, sequence, or other data structure that includes data elements of interest. A search may use heuristics to limit the search for solutions by eliminating choices that are unlikely to lead to the goal. Heuristic techniques may provide a best guess solution. A search can also include optimization. For example, a search may begin with a guess and then refine the guess incrementally until no more refinements can be made. In a further example, the disclosed systems, devices, and methods may include logic-based model and/or technique. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities can use formal or other type of logic. Logic based models may involve making inferences or deriving conclusions from a set of premises. As such, a logic based system can extend existing knowledge or create new knowledge automatically using inferences. Examples of the types of logic that can be utilized include propositional or sentential logic that comprises logic of statements which can be true or false; first-order logic that allows the use of quantifiers and predicates and that can express facts about objects, their properties, and their relations with each other; fuzzy logic that allows degrees of truth to be represented as a value between 0 and 1 rather than simply 0 (false) or 1 (true), which can be used for uncertain reasoning; subjective logic that comprises a type of probabilistic logic that may take uncertainty and belief into account, which can be suitable for modeling and analyzing situations involving uncertainty, incomplete knowledge and different world views; and/or other types of logic. In a further example, the disclosed systems, devices, and methods may include a probabilistic model and/or technique. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities can be implemented to operate with incomplete or uncertain information where probabilities may affect outcomes. Bayesian network, among other models, is an example of a probabilistic tool used for purposes such as reasoning, learning, planning, perception, and/or others. One of ordinary skill in art will understand that the aforementioned artificial intelligence models and/or techniques are described merely as

examples of a variety of possible implementations, and that while all possible artificial intelligence models and/or techniques are too voluminous to describe, other artificial intelligence models and/or techniques known in art are within the scope of this disclosure. One of ordinary skill in art will also recognize that an intelligent system may solve a specific problem by using any model and/or technique that works such as, for example, some systems can be symbolic and logical, some can be sub-symbolic neural networks, some can be deterministic or probabilistic, some can be hierarchical, some may include searching techniques, some may include optimization techniques, while others may use other or a combination of models and/or techniques. In general, any artificial intelligence model and/or technique can be utilized that can facilitate the functionalities described herein.

Referring to Figs. 17A-17C, embodiments of interconnected Knowledge Cells 800 and updating weights of Connections 853 are illustrated. As shown for example in Fig. 17A, Knowledge Cell 800za is connected to Knowledge Cell 800zb and Knowledge Cell 800zc by Connection 853z1 and Connection 853z2, respectively. Each of Connection 853z1 and Connection 853z2 may include or be associated with occurrence count, weight, and/or other parameter or data. The number of occurrences may track or store the number of observations that a Knowledge Cell 800 was followed by another Knowledge Cell 800 indicating a connection or relationship between them. For example, Knowledge Cell 800za was followed by Knowledge Cell 800zb 10 times as indicated by the number of occurrences of Connection 853z1. Also, Knowledge Cell 800za was followed by Knowledge Cell 800zc 15 times as indicated by the number of occurrences of Connection 853z2. The weight of Connection 853z1 can be calculated or determined as the number of occurrences of Connection 853z1 divided by the sum of occurrences of all connections (i.e. Connection 853z1 and Connection 853z2, etc.) originating from Knowledge Cell 800za. Therefore, the weight of Connection 853z1 can be calculated or determined as 10/(10+15)=0.4, for example. Also, the weight of Connection 853z2 can be calculated or determined as 15/(10+15)=0.6, for example. Therefore, the sum of weights of Connection 853z1, Connection 853z2, and/or any other Connections 853 originating from Knowledge Cell 800za may equal to 1 or 100%. As shown for example in Fig. 17B, in the case that Knowledge Cell 800zd is inserted and an observation is made that Knowledge Cell 800zd follows Knowledge Cell 800za, Connection 853z3 can be created between Knowledge Cell 800za and Knowledge Cell 800zd. The occurrence count of Connection 853z3 can be set to 1 and weight determined as 1/(10+15+1)=0.038. The weights of all other connections (i.e. Connection 853z1, Connection 853z2, etc.) originating from Knowledge Cell 800za may be updated to account for the creation of Connection 853z3. Therefore, the weight of Connection 853z1 can be updated as 10/(10+15+1)=0.385. The weight of Connection 853z2 can also be updated as 15/(10+15+1)=0.577. As shown for example in Fig. 17C, in the case that an additional occurrence of Connection 853z1 is observed (i.e. Knowledge Cell 800zb followed Knowledge Cell 800za, etc.), occurrence count of Connection 853z1 and weights of all connections (i.e. Connection 853z1, Connection 853z2, and Connection 853z3, etc.) originating from Knowledge Cell 800za may be updated to account for this observation. The occurrence count of Connection 853z1 can be increased by 1 and its weight updated as 11/(11+15+1)=0.407. The weight of Connection 853z2 can also be updated as 15/(11+15+1)=0.556. The weight of Connection 853z3 can also be updated as 1/(11+15+1)=0.037.

Referring to Fig. 18, an embodiment of learning Knowledge Cells 800 comprising one or more Digital Pictures 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Collection of Knowledge Cells 530d is illustrated. Collection of Knowledge Cells 530d comprises the functionality for storing any number of Knowledge Cells 800. In some aspects, Knowledge Cells 800 may be stored into or applied onto Collection of

Knowledge Cells 530d in a learning or training process. In effect, Collection of Knowledge Cells 530d may store Knowledge Cells 800 that can later be used to enable autonomous Device 98 operation. In some embodiments, Knowledge Structuring Unit 520 structures or generates Knowledge Cells 800 as previously described and the system applies them onto Collection of Knowledge Cells 530d, thereby implementing learning Device's 98 operation in various visual surroundings. The term apply or applying may refer to storing, copying, inserting, updating, or other similar action, therefore, these terms may be used interchangeably herein depending on context. The system can perform Similarity Comparisons 125 (later described) of a newly structured Knowledge Cell 800 from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Collection of Knowledge Cells 530d. If a substantially similar Knowledge Cell 800 is not found in Collection of Knowledge Cells 530d, the system may insert (i.e. copy, store, etc.) the Knowledge Cell 800 from Knowledge Structuring Unit 520 into Collection of Knowledge Cells 530d, for example. On the other hand, if a substantially similar Knowledge Cell 800 is found in Collection of Knowledge Cells 530d, the system may optionally omit inserting the Knowledge Cell 800 from Knowledge Structuring Unit 520 as inserting a substantially similar Knowledge Cell 800 may not add much or any additional knowledge to the Collection of Knowledge Cells 530d, for example. Also, inserting a substantially similar Knowledge Cell 800 can optionally be omitted to save storage resources and limit the number of Knowledge Cells 800 that may later need to be processed or compared. Any features, functionalities, and embodiments of Similarity Comparison 125, importance index (later described), similarity index (later described), and/or other disclosed elements can be utilized to facilitate determination of substantial or other similarity and whether to insert a newly structured Knowledge Cell 800 into Collection of Knowledge Cells 530d.

For example, the system can perform Similarity Comparisons 125 (later described) of Knowledge Cell 800ba from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Collection of Knowledge Cells 530d. In the case that a substantially similar match is found between Knowledge Cell 800ba and any of the Knowledge Cells 800 in Collection of Knowledge Cells 530d, the system may perform no action. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bb from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Collection of Knowledge Cells 530d. In the case that a substantially similar match is not found, the system may insert a new Knowledge Cell 800 into Collection of Knowledge Cells 530d and copy Knowledge Cell 800bb into the inserted new Knowledge Cell 800. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bc from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Collection of Knowledge Cells 530d. In the case that a substantially similar match is found between Knowledge Cell 800bc and any of the Knowledge Cells 800 in Collection of Knowledge Cells 530d, the system may perform no action. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bd from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Collection of Knowledge Cells 530d. In the case that a substantially similar match is not found, the system may insert a new Knowledge Cell 800 into Collection of Knowledge Cells 530d and copy Knowledge Cell 800bd into the inserted new Knowledge Cell 800. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800be from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Collection of Knowledge Cells 530d. In the case that a substantially similar match is not found, the system may insert a new Knowledge Cell 800 into Collection of Knowledge Cells 530d and copy Knowledge Cell 800be into the inserted new Knowledge Cell 800. Applying any additional Knowledge Cells 800 from Knowledge Structuring Unit 520 onto Collection of Knowledge Cells 530d follows similar logic or process as the above-described.

Referring to Fig. 19, an embodiment of learning Knowledge Cells 800 comprising one or more Digital Pictures 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Neural Network 530a is illustrated. Neural Network 530a includes a number of neurons or Nodes 852 interconnected by Connections 853 as previously described. Knowledge Cells 800 are shown instead of Nodes 852 to simplify the illustration as Node 852 includes a Knowledge Cell 800, for example. Therefore, Knowledge Cells 800 and Nodes 852 can be used interchangeably herein depending on context. It should be noted that Node 852 may include other elements and/or functionalities instead of or in addition to Knowledge Cell 800. In some aspects, Knowledge Cells 800 may be stored into or applied onto Neural Network 530a individually or collectively in a learning or training process. In some designs, Neural Network 530a comprises a number of Layers 854 each of which may include one or more Knowledge Cells 800. Knowledge Cells 800 in successive Layers 854 can be connected by Connections 853. Connection 853 may include or be associated with occurrence count, weight, and/or other parameter or data as previously described. Neural Network 530a may include any number of Layers 854 comprising any number of Knowledge Cells 800. In some aspects, Neural Network 530a may store Knowledge Cells 800 interconnected by Connections 853 where following a path through the Neural Network 530a can later be used to enable autonomous Device 98 operation. It should be understood that, in some embodiments, Knowledge Cells 800 in one Layer 854 of Neural Network 530a need not be connected only with Knowledge Cells 800 in a successive Layer 854, but also in any other Layer 854, thereby creating shortcuts (i.e. shortcut Connections 853, etc.) through Neural Network 530a. A Knowledge Cell 800 can also be connected to itself such as, for example, in recurrent neural networks. In general, any Knowledge Cell 800 can be connected with any other Knowledge Cell 800 anywhere else in Neural Network 530a. In further embodiments, back-propagation of any data or information can be implemented. In one example, back-propagation of similarity (i.e. similarity index, etc.) of compared Knowledge Cells 800 in a path through Neural Network 530a can be implemented. In another example, back-propagation of errors can be implemented. Such back-propagations can then be used to adjust occurrence counts and/or weights of Connections 853 for better future predictions, for example. Any other back-propagation can be implemented for other purposes. Any combination of Nodes 852 (i.e. Nodes 852 comprising Knowledge Cells 800, etc.), Connections 853, Layers 854, and/or other elements or techniques can be implemented in alternate embodiments. Neural Network 530a may include any type or form of a neural network known in art such as a feed-forward neural network, a back-propagating neural network, a recurrent neural network, a convolutional neural network, deep neural network, and/or others including a custom neural network.

In some embodiments, Knowledge Structuring Unit 520 structures or generates Knowledge Cells 800 and the system applies them onto Neural Network 530a, thereby implementing learning Device's 98 operation in various visual surroundings. The system can perform Similarity Comparisons 125 (later described) of a Knowledge Cell 800 from Knowledge Structuring Unit 520 with Knowledge Cells 800 in a corresponding Layer 854 of Neural Network 530a. If a substantially similar Knowledge Cell 800 is not found in the corresponding Layer 854 of Neural Network 530a, the system may insert (i.e. copy, store, etc.) the Knowledge Cell 800 from Knowledge Structuring Unit 520 into the corresponding Layer 854 of Neural Network 530a, and create a Connection 853 to the inserted Knowledge Cell 800 from a Knowledge Cell 800 in a prior Layer 854 including assigning an occurrence count to the new Connection 853, calculating a weight of the new Connection 853, and updating any other Connections 853 originating from the Knowledge Cell 800 in the prior Layer 854. On the other hand, if a substantially similar Knowledge Cell 800 is found

in the corresponding Layer 854 of Neural Network 530a, the system may update occurrence count and weight of Connection 853 to that Knowledge Cell 800 from a Knowledge Cell 800 in a prior Layer 854, and update any other Connections 853 originating from the Knowledge Cell 800 in the prior Layer 854.

For example, the system can perform Similarity Comparisons 125 (later described) of Knowledge Cell 800ba from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Layer 854a of Neural Network 530a. In the case that a substantially similar match is found between Knowledge Cell 800ba and Knowledge Cell 800ea, the system may perform no action since Knowledge Cell 800ea is the initial Knowledge Cell 800. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bb from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Layer 854b of Neural Network 530a. In the case that a substantially similar match is found between Knowledge Cell 800bb and Knowledge Cell 800eb, the system may update occurrence count and weight of Connection 853e1 between Knowledge Cell 800ea and Knowledge Cell 800eb, and update weights of other Connections 853 originating from Knowledge Cell 800ea as previously described. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bc from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Layer 854c of Neural Network 530a. In the case that a substantially similar match is not found, the system may insert Knowledge Cell 800ec into Layer 854c and copy Knowledge Cell 800bc into the inserted Knowledge Cell 800ec. The system may also create Connection 853e2 between Knowledge Cell 800eb and Knowledge Cell 800ec with occurrence count of 1 and weight calculated based on the occurrence count as previously described. The system may also update weights of other Connections 853 (one in this example) originating from Knowledge Cell 800eb as previously described. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bd from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Layer 854d of Neural Network 530a. In the case that a substantially similar match is not found, the system may insert Knowledge Cell 800ed into Layer 854d and copy Knowledge Cell 800bd into the inserted Knowledge Cell 800ed. The system may also create Connection 853e3 between Knowledge Cell 800ec and Knowledge Cell 800ed with occurrence count of 1 and weight of 1. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800be from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Layer 854e of Neural Network 530a. In the case that a substantially similar match is not found, the system may insert Knowledge Cell 800ee into Layer 854e and copy Knowledge Cell 800be into the inserted Knowledge Cell 800ee. The system may also create Connection 853e4 between Knowledge Cell 800ed and Knowledge Cell 800ee with occurrence count of 1 and weight of 1. Applying any additional Knowledge Cells 800 from Knowledge Structuring Unit 520 onto Neural Network 530a follows similar logic or process as the above-described.

Similarity Comparison 125 comprises the functionality for comparing or matching Knowledge Cells 800 or portions thereof, and/or other functionalities. Similarity Comparison 125 comprises the functionality for comparing or matching Digital Pictures 525 or portions thereof. Similarity Comparison 125 comprises the functionality for comparing or matching streams of Digital Pictures 525 or portions thereof. Similarity Comparison 125 comprises the functionality for comparing or matching Instruction Sets 526, Extra Info 527, text (i.e. characters, words, phrases, etc.), pictures, sounds, data, and/or other elements or portions thereof. Similarity Comparison 125 may include functions, rules, and/or logic for performing matching or comparisons and for determining that while a perfect match is not found, a similar or partial match has been found. In some aspects, Similarity Comparison 125 may include determining substantial similarity or substantial match of compared elements. In other aspects, a partial match may

include a substantial or otherwise similar match, and vice versa. Although, substantial similarity or substantial match is frequently used herein, it should be understood that any level of similarity, however high or low, may be utilized as defined by the rules (i.e. thresholds, etc.) for similarity. The rules for similarity or similar match can be defined by a user, by VSADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. In some designs, Similarity Comparison 125 comprises the functionality to automatically define appropriately strict rules for determining similarity of the compared elements. Similarity Comparison 125 can therefore set, reset, and/or adjust the strictness of the rules for finding or determining similarity of the compared elements, thereby fine tuning Similarity Comparison 125 so that the rules for determining similarity are appropriately strict. In some aspects, the rules for determining similarity may include a similarity threshold. As such, Similarity Comparison 125 can determine similarity of compared elements if their similarity exceeds a similarity threshold. In other aspects, the rules for determining similarity may include a difference threshold. As such, Similarity Comparison 125 can determine similarity of compared elements if their difference is lower than a difference threshold. In further aspects, the rules for determining similarity may include other thresholds.

In some embodiments, in determining similarity of Knowledge Cells 800, Similarity Comparison 125 can compare one or more Digital Pictures 525 or portions (i.e. regions, features, pixels, etc.) thereof from one Knowledge Cell 800 with one or more Digital Pictures 525 or portions thereof from another Knowledge Cell 800. In some aspects, total equivalence is achieved when all Digital Pictures 525 or portions thereof of the compared Knowledge Cells 800 match. If total equivalence is not found, Similarity Comparison 125 may attempt to determine substantial or other similarity. Any features, functionalities, and embodiments of the previously described Picture Recognizer 350 can be used in determining such substantial similarity.

In some embodiments where compared Knowledge Cells 800 include a single Digital Picture 525, Similarity Comparison 125 can compare Digital Picture 525 from one Knowledge Cell 800 with Digital Picture 525 from another Knowledge Cell 800 using comparison techniques for individual pictures described below. In some embodiments where compared Knowledge Cells 800 include streams of Digital Pictures 525 (i.e. motion pictures, videos, etc.), Similarity Comparison 125 can compare a stream of Digital Pictures 525 from one Knowledge Cell 800 with a stream of Digital Pictures 525 from another Knowledge Cell 800. Such comparison may include comparing Digital Pictures 525 from one Knowledge Cell 800 with corresponding (i.e. similarly positioned, temporally related, etc.) Digital Pictures 525 from another Knowledge Cell 800. In one example, a 67th Digital Picture 525 from one Knowledge Cell 800 can be compared with a 67th Digital Picture 525 from another Knowledge Cell 800. In another example, a 67th Digital Picture 525 from one Knowledge Cell 800 can be compared with a number of Digital Picture 525 around (i.e. preceding and/or following) a 67th Digital Picture 525 from another Knowledge Cell 800. This way, flexibility can be implemented in finding a substantially similar Digital Picture 525 if the Digital Pictures 525 in the compared Knowledge Cells 800 are not perfectly aligned. In other aspects, Similarity Comparison 125 can utilize Dynamic Time Warping (DTW) and/or other techniques know in art for comparing and/or aligning temporal sequences (i.e. streams of Digital Pictures 525, etc.) that may vary in time or speed. Once the corresponding (i.e. similarly positioned, temporally related, time warped/aligned, etc.) Digital Pictures 525 in the compared streams of Digital Pictures 525 are compared and their substantial similarity determined using comparison techniques for individual pictures described below, Similarity Comparison 125 can utilize a threshold for the number or percentage

of matching or substantially matching Digital Pictures 525 for determining substantial similarity of the compared Knowledge Cells 800. In some aspects, substantial similarity can be achieved when most of the Digital Pictures 525 or portions (i.e. regions, features, pixels, etc.) thereof of the compared Knowledge Cells 800 match or substantially match. In other aspects, substantial similarity can be achieved when at least a threshold number or percentage of Digital Pictures 525 or portions thereof of the compared Knowledge Cells 800 match or substantially match. Similarly, substantial similarity can be achieved when a number or percentage of matching or substantially matching Digital Pictures 525 or portions thereof of the compared Knowledge Cells 800 exceeds a threshold. In further aspects, substantial similarity can be achieved when all but a threshold number or percentage of Digital Pictures 525 or portions thereof of the compared Knowledge Cells 800 match or substantially match. Such thresholds can be defined by a user, by VSADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. In one example, substantial similarity can be achieved when at least 1, 2, 3, 4, or any other threshold number of Digital Pictures 525 or portions thereof of the compared Knowledge Cells 800 match or substantially match. Similarly, substantial similarity can be achieved when the number of matching or substantially matching Digital Pictures 525 or portions thereof of the compared Knowledge Cells 800 exceeds 1, 2, 3, 4, or any other threshold number. In another example, substantial similarity can be achieved when at least 10%, 21%, 30%, 49%, 66%, 89%, 93%, or any other percentage of Digital Pictures 525 or portions thereof of the compared Knowledge Cells 800 match or substantially match. Similarly, substantial similarity can be achieved when the percentage of matching or substantially matching Digital Pictures 525 or portions thereof of the compared Knowledge Cells 800 exceeds 10%, 21%, 30%, 49%, 66%, 89%, 93%, or any other threshold percentage. In other embodiments, substantial similarity of the compared Knowledge Cells 800 can be achieved in terms of matches or substantial matches in more important (i.e. as indicated by importance index [later described], etc.) Digital Pictures 525 or portions thereof, thereby tolerating mismatches in less important Digital Pictures 525 or portions thereof. In one example, substantial similarity can be achieved when matches or substantial matches are found with respect to more substantive Digital Pictures 525 (i.e. pictures comprising content of interest [i.e. persons, objects, etc.], etc.) or portions thereof of the compared Knowledge Cells 800, thereby tolerating mismatches in less substantive Digital Pictures 525 (i.e. pictures comprising background, insignificant content, etc.) or portions thereof. In another example, substantial similarity can be achieved when matches or substantial matches are found in earlier Digital Pictures 525 or portions thereof of the compared Knowledge Cells 800, thereby tolerating mismatches in later Digital Pictures 525 or portions thereof. In general, any importance or weight can be assigned to any Digital Picture 525 or portion thereof, and/or other elements. In some designs, Similarity Comparison 125 can be configured to omit any Digital Picture 525 or portion thereof from the comparison. In one example, less substantive Digital Pictures 525 or portions thereof can be omitted. In another example, some or all Digital Pictures 525 or portions thereof related to a specific time period can be omitted. In a further example, later Digital Pictures 525 or portions thereof can be omitted. In further embodiments, substantial similarity can be achieved taking into account the number of Digital Pictures 525 of the compared Knowledge Cells 800. For example, substantial similarity can be achieved if the number, in addition to the content, of Digital Pictures 525 of the compared Knowledge Cells 800 match or substantially match. In further embodiments, substantial similarity can be achieved taking into account the objects detected within Digital Pictures 525 and/or other features of Digital Pictures 525 of the compared Knowledge Cells 800. For example, substantial similarity can be achieved if same or similar objects are detected in Digital

Pictures 525 of the compared Knowledge Cells 800. Any features, functionalities, and embodiments of Picture Recognizer 350 can be used in such detection. In some aspects, Similarity Comparison 125 can compare the number, objects detected, and/or other features of Digital Pictures 525 as an initial check before proceeding to further detailed comparisons.

Similarity Comparison 125 can automatically adjust (i.e. increase or decrease) the strictness of the rules for determining substantial similarity of Knowledge Cells 800. In some aspects, such adjustment in strictness can be done by Similarity Comparison 125 in response to determining that total equivalence of compared Knowledge Cells 800 had not been found. Similarity Comparison 125 can keep adjusting the strictness of the rules until substantially similarity is found. All the rules or settings of substantial similarity can be set, reset, or adjusted by Similarity Comparison 125 in response to another strictness level determination. For example, Similarity Comparison 125 may attempt to find a match or substantial match in a certain percentage (i.e. 95%, etc.) of Digital Pictures 525 or portions thereof from the compared Knowledge Cells 800. If the comparison does not determine substantial similarity of compared Knowledge Cells 800, Similarity Comparison 125 may decide to decrease the strictness of the rules. In response, Similarity Comparison 125 may attempt to find fewer matching or substantially matching Digital Pictures 525 or portions thereof than in the previous attempt using stricter rules. If the comparison still does not determine substantial similarity of compared Knowledge Cells 800, Similarity Comparison 125 may further decrease (i.e. down to a certain minimum strictness or threshold, etc.) the strictness by requiring fewer Digital Pictures 525 or portions thereof to match or substantially match, thereby further increasing a chance of finding substantial similarity in compared Knowledge Cells 800. In further aspects, an adjustment in strictness can be done by Similarity Comparison 125 in response to determining that multiple substantially similar Knowledge Cells 800 had been found. Similarity Comparison 125 can keep adjusting the strictness of the rules until a best of the substantially similar Knowledge Cells 800 is found. For example, Similarity Comparison 125 may attempt to find a match or substantial match in a certain percentage (i.e. 70%, etc.) of Digital Pictures 525 or portions thereof from the compared Knowledge Cells 800. If the comparison determines a number of substantially similar Knowledge Cells 800, Similarity Comparison 125 may decide to increase the strictness of the rules to decrease the number of substantially similar Knowledge Cells 800. In response, Similarity Comparison 125 may attempt to find more matching or substantially matching Digital Pictures 525 or portions thereof in addition to the earlier found Digital Pictures 525 or portions thereof to limit the number of substantially similar Knowledge Cells 800. If the comparison still provides more than one substantially similar Knowledge Cell 800, Similarity Comparison 125 may further increase the strictness by requiring additional Digital Pictures 525 or portions thereof to match or substantially match, thereby further narrowing the number of substantially similar Knowledge Cells 800 until a best substantially similar Knowledge Cells 800 is found.

In some embodiments, in determining substantial similarity of individual Digital Pictures 525 (i.e. Digital Pictures 525 from the compared Knowledge Cells 800, etc.), Similarity Comparison 125 can compare one or more regions of one Digital Picture 525 with one or more regions of another Digital Picture 525. A region may include a collection of pixels. In some aspects, a region may include detected or recognized content of interest such as an object or person. Such region may be detected using any features, functionalities, and embodiments of Picture Recognizer 350. In other aspects, a region may include content defined using a picture segmentation technique. Examples of picture segmentation techniques include thresholding, clustering, region-growing, edge detection, curve

propagation, level sets, graph partitioning, model-based segmentation, trainable segmentation (i.e. artificial neural networks, etc.), and/or others. In further aspects, a region may include content defined using any technique. In further aspects, a region may include any arbitrary region comprising any arbitrary content. Once regions of the compared Digital Pictures 525 are known, Similarity Comparison 125 can compare the regions to determine substantial similarity of the compared Digital Pictures 525. In some aspects, total equivalence is found when all regions of one Digital Picture 525 match all regions of another Digital Picture 525. In other aspects, if total equivalence is not found, Similarity Comparison 125 may attempt to determine substantial similarity of compared Digital Pictures 525. In one example, substantial similarity can be achieved when most of the regions of the compared Digital Picture 525 match or substantially match. In another example, substantial similarity can be achieved when at least a threshold number (i.e. 1, 2, 5, 11, 39, etc.) or percentage (i.e. 38%, 63%, 77%, 84%, 98%, etc.) of regions of the compared Digital Pictures 525 match or substantially match. Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching regions of the compared Digital Pictures 525 exceeds a threshold number (i.e. 1, 2, 5, 11, 39, etc.) or a threshold percentage (i.e. 48%, 63%, 77%, 84%, 98%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or percentage of regions of the compared Digital Pictures 525 match or substantially match. Such thresholds can be defined by a user, by VSADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In further aspects, Similarity Comparison 125 can utilize the type of regions for determining substantial similarity of Digital Pictures 525. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to more substantive, larger, and/or other regions, thereby tolerating mismatches in less substantive, smaller, and/or other regions. In further aspects, Similarity Comparison 125 can utilize the importance (i.e. as indicated by importance index [later described], etc.) of regions for determining substantial similarity of Digital Pictures 525. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to more important regions such as the above described more substantive, larger, and/or other regions, thereby tolerating mismatches in less important regions such as less substantive, smaller, and/or other regions. In further aspects, Similarity Comparison 125 can omit some of the regions from the comparison in determining substantial similarity of Digital Pictures 525. In one example, isolated regions can be omitted from comparison. In another example, less substantive or smaller regions can be omitted from comparison. In general, any region can be omitted from comparison. In further aspects, Similarity Comparison 125 can focus on certain regions of interest from the compared Digital Pictures 525. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to regions comprising persons or parts (i.e. head, arm, leg, etc.) thereof, large objects, close objects, and/or other content of interest, thereby tolerating mismatches in regions comprising the background, insignificant content, and/or other content. In further aspects, Similarity Comparison 125 can detect or recognize persons or objects in the compared Digital Pictures 525 using regions. Any features, functionalities, and embodiments of Picture Recognizer 350 can be used in such detection or recognition. Once a person or object is detected in a Digital Picture 525, Similarity Comparison 125 may attempt to detect the person or object in the compared Digital Picture 525. In one example, substantial similarity can be achieved when the compared Digital Pictures 525 comprise one or more same persons or objects. In another example concerning streams of Digital

Pictures 525, substantial similarity can be achieved when the compared streams of Digital Pictures 525 comprise a detected person or object in at least a threshold number or percentage of their pictures.

Similarity Comparison 125 can automatically adjust (i.e. increase or decrease) the strictness of the rules for determining substantial similarity of Digital Pictures 525 using regions. In some aspects, such adjustment in strictness can be done by Similarity Comparison 125 in response to determining that total equivalence of compared Digital Pictures 525 had not been found. Similarity Comparison 125 can keep adjusting the strictness rules until a substantial similarity is found. All the rules or settings of substantial similarity can be set, reset, or adjusted by Similarity Comparison 125 in response to another strictness level determination. For example, Similarity Comparison 125 may attempt to find a match or substantial match in a certain percentage (i.e. 74%, etc.) of regions from the compared Digital Pictures 525. If the comparison does not determine substantial similarity of compared Digital Pictures 525, Similarity Comparison 125 may decide to decrease the strictness of the rules. In response, Similarity Comparison 125 may attempt to find fewer matching or substantially matching regions than in the previous attempt using stricter rules. If the comparison still does not determine substantial similarity of compared Digital Pictures 525, Similarity Comparison 125 may further decrease the strictness (i.e. down to a certain minimum strictness or threshold, etc.) by requiring fewer regions to match or substantially match, thereby further increasing a chance of finding substantial similarity in compared Digital Pictures 525.

Where a reference to a region is used herein it should be understood that a portion of a region or a collection of regions can be used instead of or in addition to the region. In one example, instead of or in addition to regions, individual pixels and/or features that constitute a region can be compared. In another example, instead of or in addition to regions, collections of regions can be compared. As such, any operations, rules, logic, and/or functions operating on regions similarly apply to any portion of a region and/or any collection of regions. In general, whole regions, portions of a region, and/or collections of regions, including any operations thereon, can be combined to arrive at desired results. Some or all of the above-described rules, logic, and/or techniques can be utilized alone or in combination with each other or with other rules, logic, and/or techniques. One of ordinary skill in art will recognize that other techniques known in art for determining similarity of digital pictures, streams of digital pictures, and/or other data that would be too voluminous to describe are within the scope of this disclosure.

In some embodiments, in determining substantial similarity of individual Digital Pictures 525 (i.e. Digital Pictures 525 from the compared Knowledge Cells 800, etc.), Similarity Comparison 125 can compare one or more features of one Digital Picture 525 with one or more features of another Digital Picture 525. A feature may include a collection of pixels. Some of the steps or elements in a feature oriented technique include pre-processing, feature extraction, detection/segmentation, decision-making, and/or others, or a combination thereof, each of which may include its own sub-steps or sub-elements depending on the application. Examples of features that can be used include lines, edges, ridges, corners, blobs, and/or others. Examples of feature extraction techniques include Canny, Sobe, Kayyali, Harris & Stephens et al, SUSAN, Level Curve Curvature, FAST, Laplacian of Gaussian, Difference of Gaussians, Determinant of Hessian, MSER, PCBR, Grey-level Blobs, and/or others. Once features of the compared Digital Pictures 525 are known, Similarity Comparison 125 can compare the features to determine substantial similarity. In some aspects, total equivalence is found when all features of one Digital Picture 525 match all features of another Digital Picture 525. In other aspects, if total equivalence is not found, Similarity Comparison 125 may attempt to determine substantial similarity of compared Digital Pictures 525. In one example, substantial similarity

can be achieved when most of the features of the compared Digital Picture 525 match or substantially match. In another example, substantial similarity can be achieved when at least a threshold number (i.e. 3, 22, 47, 93, 128, 431, etc.) or percentage (i.e. 49%, 53%, 68%, 72%, 95%, etc.) of features of the compared Digital Pictures 525 match or substantially match. Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching features of the compared Digital Pictures 525 exceeds a threshold number (i.e. 3, 22, 47, 93, 128, 431, etc.) or a threshold percentage (i.e. 49%, 53%, 68%, 72%, 95%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or percentage of features of the compared Digital Pictures 525 match or substantially match. Such thresholds can be defined by a user, by VSADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In further aspects, Similarity Comparison 125 can utilize the type of features for determining substantial similarity of Digital Pictures 525. In one example, substantial similarity can be achieved when matches or substantial matches are found with respect to edges, thereby tolerating mismatches in blobs. In another example, substantial similarity can be achieved when matches or substantial matches are found with respect to more substantive, larger, and/or other features, thereby tolerating mismatches in less substantive, smaller, and/or other features. In further aspects, Similarity Comparison 125 can utilize the importance (i.e. as indicated by importance index [later described], etc.) of features for determining substantial similarity of Digital Pictures 525. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to more important features such as the above described more substantive, larger, and/or other features, thereby tolerating mismatches in less important features such as less substantive, smaller, and/or other features. In further aspects, Similarity Comparison 125 can omit some of the features from the comparison in determining substantial similarity of Digital Pictures 525. In one example, isolated features can be omitted from comparison. In another example, less substantive or smaller features can be omitted from comparison. In general, any feature can be omitted from comparison. In further aspects, Similarity Comparison 125 can focus on features in certain regions of interest of the compared Digital Pictures 525. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to features in regions comprising persons or parts (i.e. head, arm, leg, etc.) thereof, large objects, close objects, and/or other objects, thereby tolerating mismatches in features of regions comprising the background, insignificant content, and/or other regions. In further aspects, Similarity Comparison 125 can detect or recognize persons or objects in the compared Digital Pictures 525. Any features, functionalities, and embodiments of Picture Recognizer 350 can be used in such detection or recognition. Once a person or object is detected in a Digital Picture 525, Similarity Comparison 125 may attempt to detect the person or object in the compared Digital Picture 525. In one example, substantial similarity can be achieved when the compared Digital Pictures 525 comprise one or more same persons or objects. In another example concerning streams of Digital Pictures 525, substantial similarity can be achieved when the compared streams of Digital Pictures 525 comprise a detected person or object in at least a threshold number or percentage of their pictures. In further aspects, Similarity Comparison 125 may include identifying and/or analyzing tiled and/or overlapping features, which can then be combined (i.e. similar to some process steps in convolutional neural networks, etc.) and compared to determine substantial similarity of Digital Pictures 525.

Similarity Comparison 125 can automatically adjust (i.e. increase or decrease) the strictness of the rules for determining substantial similarity of Digital Pictures 525 using features. In some aspects, such adjustment in

strictness can be done by Similarity Comparison 125 in response to determining that total equivalence of compared Digital Pictures 525 had not been found. Similarity Comparison 125 can keep adjusting the strictness rules until a substantial similarity is found. All the rules or settings of substantial similarity can be set, reset, or adjusted by Similarity Comparison 125 in response to another strictness level determination. For example, Similarity Comparison 125 may attempt to find a match or substantial match in a certain percentage (i.e. 89%, etc.) of features from the compared Digital Pictures 525. If the comparison does not determine substantial similarity of compared Digital Pictures 525, Similarity Comparison 125 may decide to decrease the strictness of the rules. In response, Similarity Comparison 125 may attempt to find fewer matching or substantially matching features than in the previous attempt using stricter rules. If the comparison still does not determine substantial similarity of compared Digital Pictures 525, Similarity Comparison 125 may further decrease the strictness (i.e. down to a certain minimum strictness or threshold, etc.) by requiring fewer features to match or substantially match, thereby further increasing a chance of finding substantial similarity in compared Digital Pictures 525.

Where a reference to a feature is used herein it should be understood that a portion of a feature or a collection of features can be used instead of or in addition to the feature. In one example, instead of or in addition to features, individual pixels that constitute a feature can be compared. In another example, instead of or in addition to features, collections of features can be compared. In a further example, levels of features where a feature on one level includes one or more features from another level (i.e. prior level, etc.) can be compared. As such, any operations, rules, logic, and/or functions operating on features similarly apply to any portion of a feature and/or any collection of features. In general, whole features, portions of a feature, and/or collections of features, including any operations thereon, can be combined to arrive at desired results. Some or all of the above-described rules, logic, and/or techniques can be utilized alone or in combination with each other or with other rules, logic, and/or techniques. One of ordinary skill in art will recognize that other techniques known in art for determining similarity of digital pictures, streams of digital pictures, and/or other data that would be too voluminous to describe are within the scope of this disclosure.

In some embodiments, in determining substantial similarity of individual Digital Pictures 525 (i.e. Digital Pictures 525 from the compared Knowledge Cells 800, etc.), Similarity Comparison 125 can compare pixels of one Digital Picture 525 with pixels of another Digital Picture 525. In some aspects, total equivalence is found when all pixels of one Digital Picture 525 match all pixels of another Digital Picture 525. In other aspects, if total equivalence is not found, Similarity Comparison 125 may attempt to determine substantial similarity. In one example, substantial similarity can be achieved when most of the pixels from the compared Digital Pictures 525 match or substantially match. In another example, substantial similarity can be achieved when at least a threshold number (i.e. 449, 2219, 92229, 442990, 1000028, etc.) or percentage (i.e. 39%, 45%, 58%, 72%, 92%, etc.) of pixels from the compared Digital Pictures 525 match or substantially match. Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching pixels from the compared Digital Pictures 525 exceeds a threshold number (i.e. 449, 2219, 92229, 442990, 1000028, etc.) or a threshold percentage (i.e. 39%, 45%, 58%, 72%, 92%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or percentage of pixels from the compared Digital Pictures 525 match or substantially match. Such thresholds can be defined by a user, by VSADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In further aspects, Similarity Comparison

125 can omit some of the pixels from the comparison in determining substantial similarity of Digital Pictures 525. In one example, pixels composing the background or any insignificant content can be omitted from comparison. In general, any pixel can be omitted from comparison. In further aspects, Similarity Comparison 125 can focus on pixels in certain regions of interest in determining substantial similarity of Digital Pictures 525. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to pixels in regions comprising persons or parts (i.e. head, arm, leg, etc.) thereof, large objects, close objects, and/or other content of interest, thereby tolerating mismatches in pixels in regions comprising the background, insignificant content, and/or other content.

Similarity Comparison 125 can automatically adjust (i.e. increase or decrease) the strictness of the rules for determining substantial similarity of Digital Pictures 525 using pixels. In some aspects, such adjustment in strictness can be done by Similarity Comparison 125 in response to determining that total equivalence of compared Digital Pictures 525 had not been found. Similarity Comparison 125 can keep adjusting the strictness rules until a substantial similarity is found. All the rules or settings of substantial similarity can be set, reset, or adjusted by Similarity Comparison 125 in response to another strictness level determination. For example, Similarity Comparison 125 may attempt to find a match or substantial match in a certain percentage (i.e. 77%, etc.) of pixels from the compared Digital Pictures 525. If the comparison does not determine substantial similarity of compared Digital Pictures 525, Similarity Comparison 125 may decide to decrease the strictness of the rules. In response, Similarity Comparison 125 may attempt to find fewer matching or substantially matching pixels than in the previous attempt using stricter rules. If the comparison still does not determine substantial similarity of compared Digital Pictures 525, Similarity Comparison 125 may further decrease the strictness (i.e. down to a certain minimum strictness or threshold, etc.) by requiring fewer pixels to match or substantially match, thereby further increasing a chance of finding substantial similarity in compared Digital Pictures 525.

Where a reference to a pixel is used herein it should be understood that a collection of pixels can be used instead of or in addition to the pixel. For example, instead of or in addition to pixels, collections of pixels can be compared. As such, any operations, rules, logic, and/or functions operating on pixels similarly apply to any collection of pixels. In general, pixels and/or collections of pixels, including any operations thereon, can be combined to arrive at desired results. Some or all of the above-described rules, logic, and/or techniques can be utilized alone or in combination with each other or with other rules, logic, and/or techniques. Any of the previously described features, functionalities, and embodiments of Similarity Comparison 125 for determining substantial similarity of Digital Pictures 525 using regions and/or features can similarly be used for pixels. One of ordinary skill in art will recognize that other techniques known in art for determining similarity of digital pictures, streams of digital pictures, and/or other data that would be too voluminous to describe are within the scope of this disclosure.

Other aspects or properties of digital pictures or pixels can be taken into account by Similarity Comparison 125 in digital picture comparisons. Examples of such aspects or properties include color adjustment, size adjustment, content manipulation, transparency (i.e. alpha channel, etc.), use of mask, and/or others. In some implementations, as digital pictures can be captured by various picture taking equipment, in various environments, and under various lighting conditions, Similarity Comparison 125 can adjust lighting or color of pixels or otherwise manipulate pixels before or during comparison. Lighting or color adjustment (also referred to as gray balance, neutral balance, white balance, etc.) may generally include manipulating or rebalancing the intensities of the colors

(i.e. red, green, and/or blue if RGB color model is used, etc.) of one or more pixels. For example, Similarity Comparison 125 can adjust lighting or color of all pixels of one picture to make it more comparable to another picture. Similarity Comparison 125 can also incrementally adjust the pixels such as increasing or decreasing the red, green, and/or blue pixel values by a certain amount in each cycle of comparisons in order to find a substantially similar match at one of the incremental adjustment levels. Any of the publically available, custom, or other lighting or color adjustment techniques or programs can be utilized such as color filters, color balancing, color correction, and/or others. In other implementations, Similarity Comparison 125 can resize or otherwise transform a digital picture before or during comparison. Such resizing or transformation may include increasing or decreasing the number of pixels of a digital picture. For example, Similarity Comparison 125 can increase or decrease the size of a digital picture proportionally (i.e. increase or decrease length and/or width keeping aspect ratio constant, etc.) to equate its size with the size of another digital picture. Similarity Comparison 125 can also incrementally resize a digital picture such as increasing or decreasing the size of the digital picture proportionally by a certain amount in each cycle of comparisons in order to find a substantially similar match at one of the incremental sizes. Any of the publically available, custom, or other digital picture resizing techniques or programs can be utilized such as nearest-neighbor interpolation, bilinear interpolation, bicubic interpolation, and/or others. In further implementations, Similarity Comparison 125 can manipulate content (i.e. all pixels, one or more regions, one or more depicted objects/persons, etc.) of a digital picture before or during comparison. Such content manipulation may include moving, centering, aligning, resizing, transforming, and/or otherwise manipulating content of a digital picture. For example, Similarity Comparison 125 can move, center, or align content of one picture to make it more comparable to another picture. Any of the publically available, custom, or other digital picture manipulation techniques or programs can be utilized such as pixel moving, warping, distorting, aforementioned interpolations, and/or others. In further implementations, in digital pictures comprising transparency features or functionalities, Similarity Comparison 125 can utilize a threshold for acceptable number or percentage transparency difference similar to the below-described threshold for the acceptable color difference. Alternatively, transparency can be applied to one or more pixels of a digital picture and color difference may then be determined between compared pixels taking into account the transparency related color effect. Alternatively, transparent pixels can be excluded from comparison. In further implementations, certain regions or subsets of pixels can be ignored or excluded during comparison using a mask. In general, any region or subset of a picture determined to contain no content of interest can be excluded from comparison using a mask. Examples of such regions or subsets include background, transparent or partially transparent regions, regions comprising insignificant content, or any arbitrary region or subset. Similarity Comparison 125 can perform any other pre-processing or manipulation of digital pictures or pixels before or during comparison.

In any of the comparisons involving digital pictures or pixels, Similarity Comparison 125 can utilize a threshold for acceptable number or percentage difference in determining a match for each compared pixel. A pixel in a digital picture can be encoded using various techniques such as RGB (i.e. red, green, blue), CMYK (i.e. cyan, magenta, yellow, and key [black]), binary value, hexadecimal value, numeric value, and/or others. For instance, in RGB color scheme, each of red, green, and blue colors is encoded with a value 0-255 or its binary equivalent. In one example, a threshold for acceptable difference (i.e. absolute difference, etc.) can be set at 10 for each of the three colors. Therefore, a pixel encoded as R130, G240, B50 matches or is sufficiently similar to a compared pixel

encoded as R135, G231, B57 because the differences in all three colors fall within the acceptable difference threshold (i.e. 10 in this example, etc.). Furthermore, a pixel encoded as R130, G240, B50 does not match or is not sufficiently similar to a compared pixel encoded as R143, G231, B57 because the difference in red value falls outside the acceptable difference threshold. Any other number threshold can be used such as 1, 3, 8, 15, 23, 77, 132, 197, 243, and/or others. A threshold for acceptable percentage difference can similarly be utilized such as 0.12%, 2%, 7%, 14%, 23%, 36%, 65%, and/or others. In some aspects, a threshold for acceptable number or percentage difference in red, green, and blue can be set to be different for each color. A similar difference determination can be utilized in pixels encoded in any other color scheme. The aforementioned thresholds can be defined by a user, by VSADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input.

In some embodiments, Similarity Comparison 125 can compare one or more Extra Info 527 (i.e. time information, location information, computed information, observed information, sensory information, contextual information, and/or other information, etc.) in addition to or instead of comparing Digital Pictures 525 or portions thereof in determining substantial similarity of Knowledge Cells 800. Extra Info 527 can be set to be less, equally, or more important (i.e. as indicated by importance index [later described], etc.) than Digital Pictures 525, regions, features, pixels, and/or other elements in the comparison. Since Extra Info 527 may include any contextual or other information that can be useful in determining similarity of any compared elements, Extra Info 527 can be used to enhance any of the aforementioned similarity determinations.

In some embodiments, Similarity Comparison 125 can also compare one or more Instruction Sets 526 in addition to or instead of comparing Digital Pictures 525 or portions thereof in determining substantial similarity of Knowledge Cells 800. In some aspects, Similarity Comparison 125 can compare portions of Instruction Sets 526 to determine substantial similarity of Instruction Sets 526. Similar thresholds for the number or percentage of matching portions of the compared Instruction Sets 526 can be utilized in Instruction Set 526 comparisons. Such thresholds can be defined by a user, by VSADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In other aspects, Similarity Comparison 125 can compare text (i.e. character comparison, word/phrase search/comparison, semantic comparison, etc.) or other data (i.e. bit comparison, object or data structure comparison, etc.) to determine substantial similarity of Instruction Sets 526. Any other comparison technique can be utilized in comparing Instruction Sets 526 in alternate implementations. Instruction Sets 526 can be set to be less, equally, or more important (i.e. as indicated by importance index [later described], etc.) than Digital Pictures 525, regions, features, pixels, Extra Info 527, and/or other elements in the comparison.

In some embodiments, an importance index (not shown) or other importance ranking technique can be used in any of the previously described comparisons or other processing involving elements of different importance. Importance index indicates importance of the element to or with which the index is assigned or associated. For example, importance index may indicate importance of a Knowledge Cell 800, Digital Picture 525, Instruction Set 526, Extra Info 527, region, feature, and/or other element to or with which the index is assigned or associated. In some aspects, importance index on a scale from 0 to 1 can be utilized, although, any other range can also be utilized. Importance index can be stored in or associated with the element to which the index pertains. Association of importance indexes can be implemented using a table where one column comprises elements and another column

comprises their associated importance indexes, for example. Importance indexes of various elements can be defined by a user, by VSADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. In one example, a higher Importance index can be assigned to more substantive Digital Pictures 525 (i.e. pictures comprising content of interest [i.e. persons, objects, etc.], etc.). In another example, a higher importance index can be assigned to Digital Pictures 525 that are correlated with Instruction Sets 526. Any importance index can be assigned to or associated with any element described herein. Any importance ranking technique can be utilized as or instead of importance index in alternate embodiments.

In some embodiments, Similarity Comparison 125 may generate a similarity index (not shown) for any compared elements. Similarity index indicates how well an element is matched with another element. For example, similarity index indicates how well a Knowledge Cell 800, Digital Picture 527, Instruction Set 526, Extra Info 527, region, feature, and/or other element is matched with a compared element. In some aspects, similarity index on a scale from 0 to 1 can be utilized, although, any other range can also be utilized. Similarity index can be generated by Similarity Comparison 125 whether substantial or other similarity between the compared elements is achieved or not. In one example, similarity index can be determined for a Knowledge Cell 800 based on a ratio/percentage of matched or substantially matched Digital Pictures 525 relative to the number of Digital Pictures 525 in the compared Knowledge Cell 800. Specifically, similarity index of 0.93 is determined if 93% of Digital Pictures 525 of one Knowledge Cell 800 match or substantially match Digital Pictures 525 of another Knowledge Cell 800. In some designs, importance (i.e. as indicated by importance index, etc.) of one or more Digital Pictures 525 can be included in the calculation of a weighted similarity index. Similar determination of similarity index can be implemented with Digital Pictures 525, Instruction Sets 526, Extra Info 527, regions, features, pixels, and/or other elements or portions thereof. Any combination of the aforementioned similarity index determinations or calculations can be utilized in alternate embodiments. Any similarity ranking technique can be utilized to determine or calculate similarity index in alternate embodiments.

Referring to Fig. 20, an embodiment of learning Knowledge Cells 800 comprising one or more Digital Pictures 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Neural Network 530a comprising shortcut Connections 853 is illustrated. In some designs, Knowledge Cells 800 in one Layer 854 of Neural Network 530a can be connected with Knowledge Cells 800 in any Layer 854, not only in a successive Layer 854, thereby creating shortcuts (i.e. shortcut Connections 853, etc.) through Neural Network 530a. In some aspects, creating a shortcut Connection 853 can be implemented by performing Similarity Comparisons 125 of a Knowledge Cell 800 from Knowledge Structuring Unit 520 with Knowledge Cells 800 in any Layer 854 when applying (i.e. storing, copying, etc.) the Knowledge Cell 800 from Knowledge Structuring Unit 520 onto Neural Network 530a. Once created, shortcut Connections 853 enable a wider variety of Knowledge Cells 800 to be considered when selecting a path through Neural Network 530a. In some embodiments, Knowledge Structuring Unit 520 structures or generates Knowledge Cells 800 and the system applies them onto Neural Network 530a, thereby implementing learning Device's 98 operation in various visual surroundings. The system can perform Similarity Comparisons 125 of a Knowledge Cell 800 from Knowledge Structuring Unit 520 with Knowledge Cells 800 in a corresponding and/or other Layers 854 of Neural Network 530a. If a substantially similar Knowledge Cell 800 is not found in the corresponding or other Layers 854 of Neural Network 530a, the system may insert (i.e. copy, store, etc.) the Knowledge Cell 800

from Knowledge Structuring Unit 520 into the corresponding (or another) Layer 854 of Neural Network 530a, and create a Connection 853 to the inserted Knowledge Cell 800 from a prior Knowledge Cell 800 including assigning an occurrence count to the new Connection 853, calculating a weight of the new Connection 853, and updating any other Connections 853 originating from the prior Knowledge Cell 800. On the other hand, if a substantially similar Knowledge Cell 800 is found in the corresponding or other Layers 854 of Neural Network 530a, the system may update occurrence count and weight of Connection 853 to that Knowledge Cell 800 from a prior Knowledge Cell 800, and update any other Connections 853 originating from the prior Knowledge Cell 800. Any of the previously described and/or other techniques for comparing, inserting, updating, and/or other operations on Knowledge Cells 800, Connections 853, Layers 854, and/or other elements can similarly be utilized in Neural Network 530a that comprises shortcut Connections 853.

Referring to Fig. 21, an embodiment of learning Knowledge Cells 800 comprising one or more Digital Pictures 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Graph 530b is illustrated. In some aspects, any Knowledge Cell 800 can be connected with any other Knowledge Cell 800 in Graph 530b. In other aspects, any Knowledge Cell 800 can be connected with itself and/or any other Knowledge Cell 800 in Graph 530b. In some embodiments, Knowledge Structuring Unit 520 structures or generates Knowledge Cells 800 and the system applies (i.e. store, copy, etc.) them onto Graph 530b, thereby implementing learning Device's 98 operation in various visual surroundings. The system can perform Similarity Comparisons 125 of a Knowledge Cell 800 from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Graph 530b. If a substantially similar Knowledge Cell 800 is not found in Graph 530b, the system may insert (i.e. copy, store, etc.) the Knowledge Cell 800 from Knowledge Structuring Unit 520 into Graph 530b, and create a Connection 853 to the inserted Knowledge Cell 800 from a prior Knowledge Cell 800 including assigning an occurrence count to the new Connection 853, calculating a weight of the new Connection 853, and updating any other Connections 853 originating from the prior Knowledge Cell 800. On the other hand, if a substantially similar Knowledge Cell 800 is found in Graph 530b, the system may update occurrence count and weight of Connection 853 to that Knowledge Cell 800 from a prior Knowledge Cell 800, and update any other Connections 853 originating from the prior Knowledge Cell 800. Any of the previously described and/or other techniques for comparing, inserting, updating, and/or other operations on Knowledge Cells 800, Connections 853, and/or other elements can similarly be utilized in Graph 530b.

For example, the system can perform Similarity Comparisons 125 of Knowledge Cell 800ba from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Graph 530b. In the case that a substantially similar match is not found, the system may insert Knowledge Cell 800ha into Graph 530b and copy Knowledge Cell 800ba into the inserted Knowledge Cell 800ha. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bb from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Graph 530b. In the case that a substantially similar match is found between Knowledge Cell 800bb and Knowledge Cell 800hb, the system may create Connection 853h1 between Knowledge Cell 800ha and Knowledge Cell 800hb with occurrence count of 1 and weight of 1. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bc from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Graph 530b. In the case that a substantially similar match is found between Knowledge Cell 800bc and Knowledge Cell 800hc, the system may update occurrence count and weight of Connection 853h2 between Knowledge Cell 800hb and Knowledge Cell 800hc, and update weights of other outgoing Connections 853 (one in this example) originating from Knowledge Cell 800hb as

previously described. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bd from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Graph 530b. In the case that a substantially similar match is not found, the system may insert Knowledge Cell 800hd into Graph 530b and copy Knowledge Cell 800bd into the inserted Knowledge Cell 800hd. The system may also create Connection 853h3 between Knowledge Cell 800hc and Knowledge Cell 800hd with occurrence count of 1 and weight calculated based on the occurrence count as previously described. The system may also update weights of other outgoing Connections 853 (one in this example) originating from Knowledge Cell 800hc as previously described. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800be from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Graph 530b. In the case that a substantially similar match is not found, the system may insert Knowledge Cell 800he into Graph 530b and copy Knowledge Cell 800be into the inserted Knowledge Cell 800he. The system may also create Connection 853h4 between Knowledge Cell 800hd and Knowledge Cell 800he with occurrence count of 1 and weight of 1. Applying any additional Knowledge Cells 800 from Knowledge Structuring Unit 520 onto Graph 530b follows similar logic or process as the above-described.

Referring to Fig. 22, an embodiment of learning Knowledge Cells 800 comprising one or more Digital Pictures 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Collection of Sequences 530c is illustrated. Collection of Sequences 530c comprises the functionality for storing one or more Sequences 533. Sequence 533 comprises the functionality for storing multiple Knowledge Cells 800. In some aspects, a Sequence 533 may include Knowledge Cells 800 relating to a single operation of Device 98. For example, Knowledge Structuring Unit 520 structures or generates Knowledge Cells 800 and the system applies them onto Collection of Sequences 530c, thereby implementing learning Device's 98 operation in various visual surroundings. The system can perform Similarity Comparisons 125 of Knowledge Cells 800 from Knowledge Structuring Unit 520 with corresponding Knowledge Cells 800 in Sequences 533 of Collection of Sequences 530c to find a Sequence 533 comprising Knowledge Cells 800 that are substantially similar to the Knowledge Cells 800 from Knowledge Structuring Unit 520. If Sequence 533 comprising such substantially similar Knowledge Cells 800 is not found in Collection of Sequences 530c, the system may create a new Sequence 533 comprising the Knowledge Cells 800 from Knowledge Structuring Unit 520 and insert (i.e. copy, store, etc.) the new Sequence 533 into Collection of Sequences 530c. On the other hand, if Sequence 533 comprising substantially similar Knowledge Cells 800 is found in Collection of Sequences 530c, the system may optionally omit inserting the Knowledge Cells 800 from Knowledge Structuring Unit 520 into Collection of Sequences 530c as inserting a similar Sequence 533 may not add much or any additional knowledge. This approach can save storage resources and limit the number of Knowledge Cells 800 that may later need to be processed or compared. In other aspects, a Sequence 533 may include Knowledge Cells 800 relating to a part of an operation of Device 98. Similar learning process as the above described can be utilized in such implementations. In further aspects, one or more long Sequences 533 each including Knowledge Cells 800 of multiple operations of Device 98 can be utilized. In one example, Knowledge Cells 800 of all operations can be stored in a single long Sequence 533 in which case Collection of Sequences 530c as a separate element can be omitted. In another example, Knowledge Cells 800 of multiple operations can be included in a plurality of long Sequences 533 such as hourly, daily, weekly, monthly, yearly, or other periodic or other Sequences 533. Similarity Comparisons 125 can be performed by traversing the one or more long Sequences 533 to find a match or substantially similar match. For instance, the system can perform Similarity Comparisons 125 of Knowledge Cells

800 from Knowledge Structuring Unit 520 with corresponding Knowledge Cells 800 in subsequences of a long Sequence 533 in incremental or other traversing pattern to find a subsequence comprising Knowledge Cells 800 that are substantially similar to the Knowledge Cells 800 from Knowledge Structuring Unit 520. The incremental traversing pattern may start from one end of a long Sequence 533 and move the comparison subsequence up or down one or any number of incremental Knowledge Cells 800 at a time. Other traversing patterns or methods can be employed such as starting from the middle of the Sequence 533 and subdividing the resulting sub-sequences in a recursive pattern, or any other traversing pattern or method. If a subsequence comprising substantially similar Knowledge Cells 800 is not found in the long Sequence 533, the system may concatenate or append the Knowledge Cells 800 from Knowledge Structuring Unit 520 to the long Sequence 533. In further aspects, Connections 853 can optionally be used in Sequence 533 to connect Knowledge Cells 800. For example, a Knowledge Cell 800 can be connected not only with a next Knowledge Cell 800 in the Sequence 533, but also with any other Knowledge Cell 800 in the Sequence 533, thereby creating alternate routes or shortcuts through the Sequence 533. Any number of Connections 853 connecting any Knowledge Cells 800 can be utilized. Any of the previously described and/or other techniques for comparing, inserting, updating, and/or other operations on Knowledge Cells 800, Connections 853, and/or other elements can similarly be utilized in Sequences 533 and/or Collection of Sequences 530c.

Any of the previously described data structures or arrangements of Knowledge Cells 800 such as Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, and/or others can be used alone, or in combination with each other or with other elements, in alternate embodiments. In one example, a path in Neural Network 530a or Graph 530b may include its own separate sequence of Knowledge Cells 800 that are not interconnected with Knowledge Cells 800 in other paths. In another example, a part of a path in Neural Network 530a or Graph 530b may include a sequence of Knowledge Cells 800 interconnected with Knowledge Cells 800 in other paths, whereas, another part of the path may include its own separate sequence of Knowledge Cells 800 that are not interconnected with Knowledge Cells 800 in other paths. Any other combinations or arrangements of Knowledge Cells 800 can be implemented.

Referring to Fig. 23, an embodiment of determining anticipatory Instruction Sets 526 from a single Knowledge Cell 800 is illustrated. Knowledge Cell 800 may be part of a Knowledgebase 530 (i.e. Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.) such as Collection of Knowledge Cells 530d. Decision-making Unit 540 comprises the functionality for anticipating or determining a device's operation in various visual surroundings. Decision-making Unit 540 comprises the functionality for anticipating or determining Instruction Sets 526 (i.e. anticipatory Instruction Sets 526, etc.) to be used or executed in Device's 98 autonomous operation based on incoming Digital Pictures 525 of Device's 98 visual surrounding. Decision-making Unit 540 also comprises other disclosed functionalities.

In some aspects, Decision-making Unit 540 may anticipate or determine Instruction Sets 526 (i.e. anticipatory Instruction Sets 526, etc.) for autonomous Device 98 operation by performing Similarity Comparisons 125 of incoming Digital Pictures 525 or portions thereof from Picture Capturing Apparatus 90 with Digital Pictures 525 or portions thereof from Knowledge Cells 800 in Knowledgebase 530 (i.e. Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.). A Knowledge Cell 800 includes a unit of knowledge (i.e. one or more Digital Pictures 525 correlated with any Instruction Sets 526 and/or Extra Info 527, etc.) of how Device 98 operated in a visual surrounding as previously described. When Digital

Pictures 525 or portions thereof of a similar visual surrounding are detected in the future, Decision-making Unit 540 can anticipate the Instruction Sets 526 (i.e. anticipatory Instruction Sets 526, etc.) previously learned in a similar visual surrounding, thereby enabling autonomous Device 98 operation. In some aspects, Decision-making Unit 540 can perform Similarity Comparisons 125 of incoming Digital Pictures 525 from Picture Capturing Apparatus 90 with Digital Pictures 525 from Knowledge Cells 800 in Knowledgebase 530 (i.e. Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.). If one or more substantially similar Digital Pictures 525 or portions thereof are found in a Knowledge Cell 800 from Knowledgebase 530, Instruction Sets 526 (i.e. anticipatory Instruction Sets 526, etc.) for autonomous Device 98 operation can be anticipated in Instruction Sets 526 correlated with the one or more Digital Pictures 525 from the Knowledge Cell 800. In some designs, subsequent one or more Instruction Sets 526 for autonomous Device 98 operation can be anticipated in Instruction Sets 526 correlated with subsequent Digital Pictures 525 from the Knowledge Cell 800 (or other Knowledge Cells 800), thereby anticipating not only current, but also additional future Instruction Sets 526. Although, Extra Info 527 is not shown in this and/or other figures for clarity of illustration, it should be noted that any Digital Picture 525, Instruction Set 526, and/or other element may include or be associated with Extra Info 527 and that Decision-making Unit 540 can utilize Extra Info 527 for enhanced decision making.

For example, Decision-making Unit 540 can perform Similarity Comparisons 125 of Digital Picture 525l1 or portion thereof from Picture Capturing Apparatus 90 with Digital Picture 525a1 or portion thereof from Knowledge Cell 800oa. Digital Picture 525a1 or portion thereof from Knowledge Cell 800oa may be found substantially similar. Decision-making Unit 540 can anticipate Instruction Sets 526a1-526a3 correlated with Digital Picture 525a1, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Digital Picture 525l2 or portion thereof from Picture Capturing Apparatus 90 with Digital Picture 525a2 or portion thereof from Knowledge Cell 800oa. Digital Picture 525a2 or portion thereof from Knowledge Cell 800oa may be found substantially similar. Decision-making Unit 540 can anticipate Instruction Set 526a4 correlated with Digital Picture 525a2, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Digital Picture 525l3 or portion thereof from Picture Capturing Apparatus 90 with Digital Picture 525a3 or portion thereof from Knowledge Cell 800oa. Digital Picture 525a3 or portion thereof from Knowledge Cell 800oa may be found substantially similar. Decision-making Unit 540 may not anticipate any Instruction Sets 526 since none are correlated with Digital Picture 525a3. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Digital Picture 525l4 or portion thereof from Picture Capturing Apparatus 90 with Digital Picture 525a4 or portion thereof from Knowledge Cell 800oa. Digital Picture 525a4 or portion thereof from Knowledge Cell 800oa may not be found substantially similar. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Digital Picture 525l5 or portion thereof from Picture Capturing Apparatus 90 with Digital Picture 525a5 or portion thereof from Knowledge Cell 800oa. Digital Picture 525a5 or portion thereof from Knowledge Cell 800oa may not be found substantially similar. Decision-making Unit 540 can implement similar logic or process for any additional Digital Picture 525 from Picture Capturing Apparatus 90, and so on.

It should be understood that any of the described elements and/or techniques in the foregoing example can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells 800 or elements (i.e. Digital Pictures 525, Instruction Sets 526, etc.) thereof would be affected accordingly. In one example, Extra Info 527 can be included in the Similarity Comparisons 125 as

previously described. In another example, as history of incoming Digital Pictures 525 becomes available, Decision-making Unit 540 can perform collective Similarity Comparisons 125 of the history of Digital Pictures 525 or portions thereof from Picture Capturing Apparatus 90 with subsequences of Digital Pictures 525 or portions thereof from Knowledge Cell 800. In a further example, the described comparisons in a single Knowledge Cell 800 may be performed on any number of Knowledge Cells 800 sequentially or in parallel. Parallel processors such as a plurality of Processors 11 or cores thereof can be utilized for such parallel processing. In a further example, various arrangements of Digital Pictures 525 and/or other elements in a Knowledge Cell 800 can be utilized as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to Fig. 24, an embodiment of determining anticipatory Instruction Sets 526 by traversing a single Knowledge Cell 800 is illustrated. Knowledge Cell 800 may be part of a Knowledgebase 530 (i.e. Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.) such as Collection of Knowledge Cells 530d. For example, Decision-making Unit 540 can perform Similarity Comparisons 125 of Digital Picture 525l1 or portion thereof from Picture Capturing Apparatus 90 with Digital Picture 525a1 or portion thereof from Knowledge Cell 800oa. Digital Picture 525a1 or portion thereof from Knowledge Cell 800oa may not be found substantially similar. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Digital Picture 525l1 or portion thereof from Picture Capturing Apparatus 90 with Digital Picture 525a2 or portion thereof from Knowledge Cell 800oa. Digital Picture 525a2 or portion thereof from Knowledge Cell 800oa may not be found substantially similar. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Digital Picture 525l1 or portion thereof from Picture Capturing Apparatus 90 with Digital Picture 525a3 or portion thereof from Knowledge Cell 800oa. Digital Picture 525a3 or portion thereof from Knowledge Cell 800oa may be found substantially similar. Decision-making Unit 540 may not anticipate any Instruction Sets 526 since none are correlated with Digital Picture 525a3. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Digital Picture 525l2 or portion thereof from Picture Capturing Apparatus 90 with Digital Picture 525a4 or portion thereof from Knowledge Cell 800oa. Digital Picture 525a4 or portion thereof from Knowledge Cell 800oa may be found substantially similar. Decision-making Unit 540 can anticipate Instruction Sets 526a5-526a6 correlated with Digital Picture 525a4, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Digital Picture 525l3 or portion thereof from Picture Capturing Apparatus 90 with Digital Picture 525a5 or portion thereof from Knowledge Cell 800oa. Digital Picture 525a5 or portion thereof from Knowledge Cell 800oa may be found substantially similar. Decision-making Unit 540 may not anticipate any Instruction Sets 526 since none are correlated with Digital Picture 525a5. Decision-making Unit 540 can implement similar logic or process for any additional Digital Pictures 525 from Picture Capturing Apparatus 90, and so on.

It should be understood that any of the described elements and/or techniques in the foregoing example can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells 800 or elements (i.e. Digital Pictures 525, Instruction Sets 526, etc.) thereof would be affected accordingly. In one example, Extra Info 527 can be included in the Similarity Comparisons 125 as previously described. In another example, as history of incoming Digital Pictures 525 becomes available, Decision-making Unit 540 can perform collective Similarity Comparisons 125 of the history of Digital Pictures 525 or portions

thereof from Picture Capturing Apparatus 90 with subsequences of Digital Pictures 525 or portions thereof from Knowledge Cell 800. In a further example, traversing may be performed in incremental traversing pattern such as starting from one end of Knowledge Cell 800 and moving the comparison subsequence up or down the list one or any number of incremental Digital Pictures 525 at a time. Other traversing patterns or methods can be employed such as starting from the middle of the Knowledge Cell 800 and subdividing the resulting subsequence in a recursive pattern, or any other traversing pattern or method. In a further example, the described traversing of a single Knowledge Cell 800 may be performed on any number of Knowledge Cells 800 sequentially or in parallel. Parallel processors such as a plurality of Processors 11 or cores thereof can be utilized for such parallel processing. In a further example, various arrangements of Digital Pictures 525 and/or other elements in a Knowledge Cell 800 can be utilized as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to Fig. 25, an embodiment of determining anticipatory Instruction Sets 526 using collective similarity comparisons is illustrated. For example, Decision-making Unit 540 can perform Similarity Comparisons 125 of Digital Picture 525l1 or portion thereof from Picture Capturing Apparatus 90 with corresponding Digital Pictures 525 or portions thereof from Knowledge Cells 800 in Collection of Knowledge Cells 530d. Digital Picture 525c1 or portion thereof from Knowledge Cell 800rc may be found substantially similar with highest similarity. Decision-making Unit 540 can anticipate any Instruction Sets 526 (not shown) correlated with Digital Picture 525c1, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform collective Similarity Comparisons 125 of Digital Pictures 525l1-525l2 or portions thereof from Picture Capturing Apparatus 90 with corresponding Digital Pictures 525 or portions thereof from Knowledge Cells 800 in Collection of Knowledge Cells 530d. Digital Pictures 525c1-525c2 or portions thereof from Knowledge Cell 800rc may be found substantially similar with highest similarity. Decision-making Unit 540 can anticipate any Instruction Sets 526 (not shown) correlated with Digital Picture 525c2, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform collective Similarity Comparisons 125 of Digital Pictures 525l1-525l3 or portions thereof from Picture Capturing Apparatus 90 with corresponding Digital Pictures 525 or portions thereof from Knowledge Cells 800 in Collection of Knowledge Cells 530d. Digital Pictures 525d1-525d3 or portions thereof from Knowledge Cell 800rd may be found substantially similar with highest similarity. Decision-making Unit 540 can anticipate any Instruction Sets 526 (not shown) correlated with Digital Picture 525d3, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform collective Similarity Comparisons 125 of Digital Pictures 525l1-525l4 or portions thereof from Picture Capturing Apparatus 90 with corresponding Digital Pictures 525 or portions thereof from Knowledge Cells 800 in Collection of Knowledge Cells 530d. Digital Pictures 525d1-525d4 or portions thereof from Knowledge Cell 800rd may be found substantially similar with highest similarity. Decision-making Unit 540 can anticipate any Instruction Sets 526 (not shown) correlated with Digital Picture 525d4, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform collective Similarity Comparisons 125 of Digital Pictures 525l1-525l5 or portions thereof from Picture Capturing Apparatus 90 with corresponding Digital Pictures 525 or portions thereof from Knowledge Cells 800 in Collection of Knowledge Cells 530d. Digital Pictures 525d1-525d5 or portions thereof from Knowledge Cell 800rd may be found substantially similar with highest similarity. Decision-making Unit 540 can anticipate any Instruction Sets 526 (not shown) correlated with Digital Picture 525d5,

thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can implement similar logic or process for any additional Digital Picture 525 from Picture Capturing Apparatus 90, and so on.

In some embodiments, various elements and/or techniques can be utilized in the aforementioned similarity determinations with respect to collectively compared Digital Pictures 525 and/or other elements. In some aspects, similarity of collectively compared Digital Pictures 525 can be determined based on similarities or similarity indexes of the individually compared Digital Pictures 525. In one example, an average of similarities or similarity indexes of individually compared Digital Pictures 525 can be used to determine similarity of collectively compared Digital Pictures 525. In another example, a weighted average of similarities or similarity indexes of individually compared Digital Pictures 525 can be used to determine similarity of collectively compared Digital Pictures 525. For instance, to affect the weighting of collective similarity, a higher weight or importance (i.e. importance index, etc.) can be assigned to the similarities or similarity indexes of some (i.e. more substantive, etc.) Digital Pictures 525 and lower for other (i.e. less substantive, etc.) Digital Pictures 525. Any other higher or lower weight or importance assignment can be implemented. In other aspects, any of the previously described or other thresholds for substantial similarity of individually compared elements can be similarly utilized for collectively compared elements. In one example, substantial similarity of collectively compared Digital Pictures 525 can be achieved when their collective similarity or similarity index exceeds a similarity threshold. In another example, substantial similarity of collectively compared Digital Pictures 525 can be achieved when at least a threshold number or percentage of Digital Pictures 525 or portions thereof of the collectively compared Digital Pictures 525 match or substantially match. Similarly, substantial similarity of collectively compared Digital Pictures 525 can be achieved when a number or percentage of matching or substantially matching Digital Pictures 525 or portions thereof of the collectively compared Digital Pictures 525 exceeds a threshold. Such thresholds can be defined by a user, by VSADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. Similar elements and/or techniques as the aforementioned can be used for similarity determinations of other collectively compared elements such as Instruction Sets 526, Extra Info 527, Knowledge Cells 800, and/or others. Similarity determinations of collectively compared elements may include any features, functionalities, and embodiments of Similarity Comparison 125, and vice versa.

It should be understood that any of the described elements and/or techniques in the foregoing example can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells 800 or elements (i.e. Digital Pictures 525, Instruction Sets 526, etc.) thereof would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info 527 in Similarity Comparisons 125, traversing of Knowledge Cells 800 or other elements, using history of Digital Pictures 525 or Knowledge Cells 800 for collective Similarity Comparisons 125, using various arrangements of Digital Pictures 525 and/or other elements in a Knowledge Cell 800, and/or others can similarly be utilized in this example. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to Fig. 26, an embodiment of determining anticipatory Instruction Sets 526 using Neural Network 530a is illustrated. In some aspects, determining anticipatory Instruction Sets 526 using Neural Network 530a may include selecting a path of Knowledge Cells 800 or elements (i.e. Digital Pictures 525, Instruction Sets 526, etc.)

thereof through Neural Network 530a. Decision-making Unit 540 can utilize various elements and/or techniques for selecting a path through Neural Network 530a. Although, these elements and/or techniques are described using Neural Network 530a below, they can similarly be used in any Knowledgebase 530 (i.e. Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.) where applicable.

In some embodiments, Decision-making Unit 540 can utilize similarity index in selecting Knowledge Cells 800 or elements (i.e. Digital Pictures 525, Instruction Sets 526, etc.) thereof in a path through Neural Network 530a. For instance, similarity index may indicate how well one or more Digital Pictures 525 or portions thereof are matched with one or more other Digital Pictures 525 or portions thereof as previously described. In one example, Decision-making Unit 540 may select a Knowledge Cell 800 comprising one or more Digital Pictures 525 with highest similarity index even if Connection 853 pointing to that Knowledge Cell 800 has less than the highest weight. Therefore, similarity index or other such element or parameter can override or disregard the weight of a Connection 853 or other element. In another example, Decision-making Unit 540 may select a Knowledge Cell 800 comprising one or more Digital Pictures 525 whose similarity index is higher than or equal to a weight of Connection 853 pointing to that Knowledge Cell 800. In a further example, Decision-making Unit 540 may select a Knowledge Cell 800 comprising one or more Digital Pictures 525 whose similarity index is lower than or equal to a weight of Connection 853 pointing to that Knowledge Cell 800. Similarity index can be set to be more, less, or equally important than a weight of a Connection 853.

In other embodiments, Decision-making Unit 540 can utilize Connections 853 in selecting Knowledge Cells 800 or elements (i.e. Digital Pictures 525, Instruction Sets 526, etc.) thereof in a path through Neural Network 530a. In some aspects, Decision-making Unit 540 can take into account weights of Connections 853 among the interconnected Knowledge Cells 800 in choosing from which Knowledge Cell 800 to compare one or more Digital Pictures 525 first, second, third, and so on. Specifically, for instance, Decision-making Unit 540 can perform Similarity Comparison 125 with one or more Digital Pictures 525 from Knowledge Cell 800 pointed to by the highest weight Connection 853 first, Digital Pictures 525 from Knowledge Cell 800 pointed to by the second highest weight Connection 853 second, and so on. In other aspects, Decision-making Unit 540 can stop performing Similarity Comparisons 125 as soon as it finds one or more substantially similar Digital Pictures 525 in an interconnected Knowledge Cell 800. In further aspects, Decision-making Unit 540 may only follow the highest weight Connection 853 to arrive at a Knowledge Cell 800 comprising one or more Digital Pictures 525 to be compared, thereby disregarding Connections 853 with less than the highest weight. In further aspects, Decision-making Unit 540 may ignore Connections 853 and/or their weights.

In further embodiments, Decision-making Unit 540 can utilize a bias to adjust similarity index, weight of a Connection 853, and/or other element or parameter used in selecting Knowledge Cells 800 or elements (i.e. Digital Pictures 525, Instruction Sets 526, etc.) thereof in a path through Neural Network 530a. In one example, Decision-making Unit 540 may select a Knowledge Cell 800 comprising one or more Digital Pictures 525 whose similarity index multiplied by or adjusted for a bias is higher than or equal to a weight of Connection 853 pointing to that Knowledge Cell 800. In another example, Decision-making Unit 540 may select a Knowledge Cell 800 comprising one or more Digital Pictures 525 whose similarity index multiplied by or adjusted for a bias is lower than or equal to a weight of Connection 853 pointing to that Knowledge Cell 800. In a further example, bias can be used to resolve deadlock situations where similarity index is equal to a weight of a Connection 853. In some aspects, bias can be

expressed in percentages such as 0.3 percent, 1.2 percent, 25.7 percent, 79.8 percent, 99.9 percent, 100.1 percent, 155.4 percent, 298.6 percent, 1105.5 percent, and so on. For example, a bias below 100 percent decreases an element or parameter to which it is applied, a bias equal to 100 percent does not change the element or parameter to which it is applied, and a bias higher than 100 percent increases the element or parameter to which it is applied. In general, any amount of bias can be utilized. Bias can be applied to one or more of a weight of a Connection 853, similarity index, any other element or parameter, and/or all or any combination of them. Also, different biases can be applied to each of a weight of a Connection 853, similarity index, or any other element or parameter. For example, 30 percent bias can be applied to similarity index and 15 percent bias can be applied to a weight of a Connection 853. Also, different biases can be applied to various Layers 854 of Neural Network 530a, and/or other disclosed elements. Bias can be defined by a user, by VSADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input.

Any other element and/or technique can be utilized in selecting Knowledge Cells 800 or elements (i.e. Digital Pictures 525, Instruction Sets 526, etc.) thereof in a path through Neural Network 530a.

In some embodiments, Neural Network 530a may include knowledge (i.e. interconnected Knowledge Cells 800 comprising one or more Digital Pictures 525 correlated with any Instruction Sets 526 and/or Extra Info 527, etc.) of how Device 98 operated in various visual surroundings. In some aspects, determining anticipatory Instruction Sets 526 using Neural Network 530a may include selecting a path of Knowledge Cells 800 or elements (i.e. Digital Pictures 525, Instruction Sets 526, etc.) thereof through Neural Network 530a. Individual and/or collective Similarity Comparisons 125 can be used to determine substantial similarity of the individually and/or collectively compared Digital Pictures 525 or portions thereof. Substantial similarity may be used primarily for selecting a path through Neural Network 530a, whereas, weight of any Connection 853 may be used secondarily or not at all.

For example, Decision-making Unit 540 can perform Similarity Comparisons 125 of Digital Pictures 525a1-525an or portions thereof from Picture Capturing Apparatus 90 with Digital Pictures 525 or portions thereof from one or more Knowledge Cells 800 in Layer 854a (or any other one or more Layers 854, etc.). Digital Pictures 525 or portions thereof from Knowledge Cell 800ta may be found collectively substantially similar with highest similarity. As the comparisons of individual Digital Pictures 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Digital Pictures 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Digital Pictures 525b1-525bn or portions thereof from Picture Capturing Apparatus 90 with Digital Pictures 525 or portions thereof from one or more Knowledge Cells 800 in Layer 854b interconnected with Knowledge Cell 800ta. Digital Pictures 525 or portions thereof from Knowledge Cell 800tb may be found collectively substantially similar with highest similarity, thus, Decision-making Unit 540 may follow Connection 853t1 disregarding its less than highest weight. As the comparisons of individual Digital Pictures 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Digital Pictures 525 as previously described, thereby enabling autonomous Device 98 operation. Since Connection 853t2 is the only connection from Knowledge Cell 800tb, Decision-making Unit 540 may follow Connection 853t2 and perform Similarity Comparisons 125 of Digital Pictures 525c1-525cn or portions thereof from Picture Capturing Apparatus 90 with Digital Pictures 525 or portions thereof from Knowledge Cell 800tc in Layer 854c. Digital Pictures 525 or portions thereof from Knowledge Cell 800tc may be found

collectively substantially similar. As the comparisons of individual Digital Pictures 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Digital Pictures 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Digital Pictures 525d1-525dn or portions thereof from Picture Capturing Apparatus 90 with Digital Pictures 525 or portions thereof from one or more Knowledge Cells 800 in Layer 854d interconnected with Knowledge Cell 800tc. Digital Pictures 525 or portions thereof from Knowledge Cell 800td may be found collectively substantially similar with highest similarity, thus, Decision-making Unit 540 may follow Connection 853t3. As the comparisons of individual Digital Pictures 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Digital Pictures 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Digital Pictures 525e1-525en or portions thereof from Picture Capturing Apparatus 90 with Digital Pictures 525 or portions thereof from one or more Knowledge Cells 800 in Layer 854e interconnected with Knowledge Cell 800td. Digital Pictures 525 or portions thereof from Knowledge Cell 800te may be found collectively substantially similar with highest similarity, thus, Decision-making Unit 540 may follow Connection 853t4. As the comparisons of individual Digital Pictures 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Digital Pictures 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can implement similar logic or process for any additional Digital Pictures 525 from Picture Capturing Apparatus 90, and so on.

The foregoing exemplary embodiment provides an example of utilizing a combination of collective Similarity Comparisons 125, individual Similarity Comparisons 125, Connections 853, and/or other elements or techniques. It should be understood that any of these elements and/or techniques can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells 800 or elements (i.e. Digital Pictures 525, Instruction Sets 526, etc.) thereof in a path through Neural Network 530a would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info 527 in Similarity Comparisons 125, traversing of Knowledge Cells 800 or other elements, using history of Digital Pictures 525 or Knowledge Cells 800 for collective Similarity Comparisons 125, using various arrangements of Digital Pictures 525 and/or other elements in a Knowledge Cell 800, and/or others can similarly be utilized in this example. These elements and/or techniques can similarly be utilized in Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, and/or other data structures or arrangements. In some aspects, instead of anticipating Instruction Sets 526 correlated with substantially similar individual Digital Pictures 525, Decision-making Unit 540 can anticipate instruction Sets 526 correlated with substantially similar streams of Digital Pictures 525. In other aspects, any time that substantial similarity or other similarity threshold is not achieved in compared Digital Pictures 525 or portions thereof of any of the Knowledge Cells 800, Decision-making Unit 540 can decide to look for a substantially or otherwise similar Digital Pictures 525 or portions thereof in Knowledge Cells 800 elsewhere in Neural Network 530a such as in any Layer 854 subsequent to a current Layer 854, in the first Layer 854, in the entire Neural Network 530a, and/or others, even if such Knowledge Cell 800 may be unconnected with a prior Knowledge Cell 800. It should be noted that any of Digital Pictures 525a1-525an, Digital Pictures 525b1-525bn, Digital Pictures 525c1-525cn, Digital Pictures 525d1-

525dn, Digital Pictures 525e1-525en, etc. may include one Digital Picture 525 or a stream of Digital Pictures 525. It should also be noted that any Knowledge Cell 800 may include one Digital Picture 525 or a stream of Digital Pictures 525 as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to Fig. 27, an embodiment of determining anticipatory Instruction Sets 526 using Graph 530b is illustrated. Graph 530b may include knowledge (i.e. interconnected Knowledge Cells 800 comprising one or more Digital Pictures 525 correlated with any Instruction Sets 526 and/or Extra Info 527, etc.) of how Device 98 operated in various visual surroundings. In some aspects, determining anticipatory Instruction Sets 526 using Graph 530b may include selecting a path of Knowledge Cells 800 or elements (i.e. Digital Pictures 525, Instruction Sets 526, etc.) thereof through Graph 530b. Individual and/or collective Similarity Comparisons 125 can be used to determine substantial similarity of the individually and/or collectively compared Digital Pictures 525 or portions thereof. Substantial similarity may be used primarily for selecting a path through Graph 530b, whereas, weight of any Connection 853 may be used secondarily or not at all.

For example, Decision-making Unit 540 can perform Similarity Comparisons 125 of Digital Pictures 525a1-525an or portions thereof from Picture Capturing Apparatus 90 with Digital Pictures 525 or portions thereof from one or more Knowledge Cells 800 in Graph 530b. Digital Pictures 525 or portions thereof from Knowledge Cell 800ua may be found collectively substantially similar with highest similarity. As the comparisons of individual Digital Pictures 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Digital Pictures 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Digital Pictures 525b1-525bn or portions thereof from Picture Capturing Apparatus 90 with Digital Pictures 525 or portions thereof from one or more Knowledge Cells 800 in Graph 530b interconnected with Knowledge Cell 800ua by outgoing Connections 853. Digital Pictures 525 or portions thereof from Knowledge Cell 800ub may be found collectively substantially similar with highest similarity, thus, Decision-making Unit 540 may follow Connection 853u1 disregarding its less than highest weight. As the comparisons of individual Digital Pictures 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Digital Pictures 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Digital Pictures 525c1-525cn or portions thereof from Picture Capturing Apparatus 90 with Digital Pictures 525 or portions thereof from one or more Knowledge Cells 800 in Graph 530b interconnected with Knowledge Cell 800ub by outgoing Connections 853. Digital Pictures 525 or portions thereof from Knowledge Cell 800uc may be found collectively substantially similar with highest similarity, thus, Decision-making Unit 540 may follow Connection 853u2 disregarding its less than highest weight. As the comparisons of individual Digital Pictures 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Digital Pictures 525 as previously described, thereby enabling autonomous Device 98 operation. Since Connection 853u3 is the only connection from Knowledge Cell 800uc, Decision-making Unit 540 may follow Connection 853u3 and perform Similarity Comparisons 125 of Digital Pictures 525d1-525dn or portions thereof from Picture Capturing Apparatus 90 with Digital Pictures 525 or portions thereof from Knowledge Cell 800ud in Graph 530b. Digital

Pictures 525 or portions thereof from Knowledge Cell 800ud may be found collectively substantially similar. As the comparisons of individual Digital Pictures 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Digital Pictures 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Digital Pictures 525e1-525en or portions thereof from Picture Capturing Apparatus 90 with Digital Pictures 525 or portions thereof from one or more Knowledge Cells 800 in Graph 530b interconnected with Knowledge Cell 800ud by outgoing Connections 853. Digital Pictures 525 or portions thereof from Knowledge Cell 800ue may be found collectively substantially similar with highest similarity, thus, Decision-making Unit 540 may follow Connection 853u4. As the comparisons of individual Digital Pictures 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Digital Pictures 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can implement similar logic or process for any additional Digital Pictures 525 from Activity Detector 160, and so on.

The foregoing exemplary embodiment provides an example of utilizing a combination of collective Similarity Comparisons 125, individual Similarity Comparisons 125, Connections 853, and/or other elements or techniques. It should be understood that any of these elements and/or techniques can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells 800 or elements (i.e. Digital Pictures 525, Instruction Sets 526, etc.) thereof in a path through Graph 530b would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info 527 in Similarity Comparisons 125, traversing of Knowledge Cells 800 or other elements, using history of Digital Pictures 525 or Knowledge Cells 800 in collective Similarity Comparisons 125, using various arrangements of Digital Pictures 525 and/or other elements in a Knowledge Cell 800, and/or others can similarly be utilized in this example. These elements and/or techniques can similarly be utilized in Neural Network 530a, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, and/or other data structures or arrangements. In some aspects, instead of anticipating Instruction Sets 526 correlated with substantially similar individual Digital Pictures 525, Decision-making Unit 540 can anticipate instruction Sets 526 correlated with substantially matching streams of Digital Pictures 525. In other aspects, any time that substantial similarity or other similarity threshold is not achieved in compared Digital Pictures 525 or portions thereof of any of the Knowledge Cells 800, Decision-making Unit 540 can decide to look for a substantially or otherwise similar Digital Pictures 525 or portions thereof in Knowledge Cells 800 elsewhere in Graph 530b even if such Knowledge Cell 800 may be unconnected with a prior Knowledge Cell 800. It should be noted that any of Digital Pictures 525a1-525an, Digital Pictures 525b1-525bn, Digital Pictures 525c1-525cn, Digital Pictures 525d1-525dn, Digital Pictures 525e1-525en, etc. may include one Digital Picture 525 or a stream of Digital Pictures 525. It should also be noted that any Knowledge Cell 800 may include one Digital Picture 525 or a stream of Digital Pictures 525 as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to Fig. 28, an embodiment of determining anticipatory Instruction Sets 526 using Collection of Sequences 530c is illustrated. Collection of Sequences 530c may include knowledge (i.e. sequences of Knowledge

Cells 800 comprising one or more Digital Pictures 525 correlated with any Instruction Sets 526 and/or Extra Info 527, etc.) of how Device 98 operated in various visual surroundings. In some aspects, determining anticipatory Instruction Sets 526 for autonomous Device 98 operation using Collection of Sequences 530c may include selecting a Sequence 533 of Knowledge Cells 800 or elements (i.e. Digital Pictures 525, Instruction Sets 526, etc.) thereof from Collection of Sequences 530c. Individual and/or collective Similarity Comparisons 125 can be used to determine substantial similarity of the individually and/or collectively compared Digital Pictures 525 or portions thereof.

For example, Decision-making Unit 540 can perform Similarity Comparisons 125 of Digital Pictures 525a1-525an or portions thereof from Picture Capturing Apparatus 90 with Digital Pictures 525 or portions thereof from corresponding Knowledge Cells 800 in one or more Sequences 533 of Collection of Sequences 530c. Digital Pictures 525 or portions thereof from Knowledge Cell 800ca in Sequence 533wc may be found collectively substantially similar with highest similarity. As the comparisons of individual Digital Pictures 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Digital Pictures 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Digital Pictures 525a1-525an and 525b1-525bn or portions thereof from Picture Capturing Apparatus 90 with Digital Pictures 525 or portions thereof from corresponding Knowledge Cells 800 in Sequences 533 of Collection of Sequences 530c. Digital Pictures 525 or portions thereof from Knowledge Cells 800ca-800cb in Sequence 533wc may be found collectively substantially similar with highest similarity. As the comparisons of individual Digital Pictures 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Digital Pictures 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Digital Pictures 525a1-525an, 525b1-525bn, and 525c1-525cn or portions thereof from Picture Capturing Apparatus 90 with Digital Pictures 525 or portions thereof from corresponding Knowledge Cells 800 in Sequences 533 of Collection of Sequences 530c. Digital Pictures 525 or portions thereof from Knowledge Cells 800da-800dc in Sequence 533wd may be found substantially similar with highest similarity. As the comparisons of individual Digital Pictures 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Digital Pictures 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Digital Pictures 525a1-525an, 525b1-525bn, 525c1-525cn, and 525d1-525dn or portions thereof from Picture Capturing Apparatus 90 with Digital Pictures 525 or portions thereof from corresponding Knowledge Cells 800 in Sequences 533 of Collection of Sequences 530c. Digital Pictures 525 or portions thereof from Knowledge Cells 800da-800dd in Sequence 533wd may be found substantially similar with highest similarity. As the comparisons of individual Digital Pictures 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Digital Pictures 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Digital Pictures 525a1-525an, 525b1-525bn, 525c1-525cn, 525d1-525dn, and 525e1-525en or portions thereof from Picture Capturing Apparatus 90 with Digital Pictures 525 or portions thereof from corresponding Knowledge Cells 800 in Sequences 533 of Collection of Sequences 530c. Digital Pictures 525 or portions thereof from Knowledge Cells

800da-800de in Sequence 533wd may be found substantially similar with highest similarity. As the comparisons of individual Digital Pictures 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Digital Pictures 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can implement similar logic or process for any additional Digital Pictures 525 from Picture Capturing Apparatus 90, and so on.

In some embodiments, various elements and/or techniques can be utilized in the aforementioned substantial similarity determinations with respect to collectively compared Knowledge Cells 800 or elements (i.e. Digital Pictures 525, Extra Info 527, etc.) thereof. In some aspects, substantial similarity of collectively compared Knowledge Cells 800 or elements (i.e. Digital Pictures 525, Extra Info 527, etc.) thereof can be determined based on similarities or similarity indexes of the individually compared Knowledge Cells 800 or elements (i.e. Digital Pictures 525, Extra Info 527, etc.) thereof. In one example, an average of similarities or similarity indexes of individually compared Knowledge Cells 800 or elements (i.e. Digital Pictures 525, Extra Info 527, etc.) thereof can be used to determine similarity of collectively compared Knowledge Cells 800 or elements (i.e. Digital Pictures 525, Extra Info 527, etc.) thereof. In another example, a weighted average of similarities or similarity indexes of individually compared Knowledge Cells 800 or elements (i.e. Digital Pictures 525, Extra Info 527, etc.) thereof can be used to determine similarity of collectively compared Knowledge Cells 800 or elements (i.e. Digital Pictures 525, Extra Info 527, etc.) thereof. For instance, to affect the weighting of collective similarity, a higher weight or importance (i.e. importance index, etc.) can be assigned to the similarities or similarity indexes of some Knowledge Cells 800 or elements (i.e. Digital Pictures 525, Extra Info 527, etc.) thereof and lower for other Knowledge Cells 800 or elements (i.e. Digital Pictures 525, Extra Info 527, etc.) thereof. Any higher or lower weight or importance assignment can be implemented. In other aspects, any of the previously described or other thresholds for substantial similarity of individually compared elements can similarly be utilized for collectively compared elements. In one example, substantial similarity of collectively compared Knowledge Cells 800 or elements (i.e. Digital Pictures 525, Extra Info 527, etc.) thereof can be achieved when their collective similarity or similarity index exceeds a similarity threshold. In another example, substantial similarity of collectively compared Knowledge Cells 800 can be achieved when at least a threshold number or percentage of Digital Pictures 525 or portions thereof of the collectively compared Knowledge Cells 800 match or substantially match. Similarly, substantial similarity of collectively compared Knowledge Cells 800 can be achieved when a number or percentage of matching or substantially matching Digital Pictures 525 or portions thereof of the collectively compared Knowledge Cells 800 exceeds a threshold. Such thresholds can be defined by a user, by VSADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. Collective similarity determinations may include any features, functionalities, and embodiments of Similarity Comparison 125, and vice versa.

The foregoing exemplary embodiment provides an example of utilizing a combination of collective Similarity Comparisons 125, individual Similarity Comparisons 125, and/or other elements or techniques. It should be understood that any of these elements and/or techniques can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Sequence 533 of Knowledge Cells 800 or elements (i.e. Digital Pictures 525, Instruction Sets 526, etc.) thereof would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info 527 in Similarity Comparisons 125, traversing of Knowledge Cells 800 or other elements, using history of Digital

Pictures 525 or Knowledge Cells 800 in collective Similarity Comparisons 125, using various arrangements of Digital Pictures 525 and/or other elements in a Knowledge Cell 800, and/or others can similarly be utilized in this example. These elements and/or techniques can similarly be utilized in Neural Network 530a, Graph 530b, Collection of Knowledge Cells 530d, and/or other data structures or arrangements. In some aspects, instead of anticipating Instruction Sets 526 correlated with substantially similar individual Digital Pictures 525, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially matching streams of Digital Pictures 525. In other aspects, any time that substantial similarity or other similarity threshold is not achieved in compared Digital Pictures 525 or portions thereof of any of the Knowledge Cells 800, Decision-making Unit 540 can decide to look for a substantially or otherwise similar Digital Pictures 525 or portions thereof in Knowledge Cells 800 elsewhere in Collection of Sequences 530c such as in different Sequences 533. It should be noted that any of Digital Pictures 525a1-525an, Digital Pictures 525b1-525bn, Digital Pictures 525c1-525cn, Digital Pictures 525d1-525dn, Digital Pictures 525e1-525en, etc. may include one Digital Picture 525 or a stream of Digital Pictures 525. It should also be noted that any Knowledge Cell 800 may include one Digital Picture 525 or a stream of Digital Pictures 525 as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring now to Modification Interface 130. Modification Interface 130 comprises the functionality for modifying execution and/or functionality of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. Modification Interface 130 comprises the functionality for modifying execution and/or functionality of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element at runtime. Modification Interface 130 comprises the functionality for modifying execution and/or functionality of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element based on anticipatory Instruction Sets 526. In one example, Modification Interface 130 comprises the functionality to access, modify, and/or perform other manipulations on runtime engine/environment, virtual machine, operating system, compiler, just-in-time (JIT) compiler, interpreter, translator, execution stack, file, object, data structure, and/or other computing system elements. In another example, Modification Interface 130 comprises the functionality to access, modify, and/or perform other manipulations on memory, storage, bus, interfaces, and/or other computing system elements. In a further example, Modification Interface 130 comprises the functionality to access, modify, and/or perform other manipulations on Processor 11 registers and/or other Processor 11 elements. In a further example, Modification Interface 130 comprises the functionality to access, modify, and/or perform other manipulations on inputs and/or outputs of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. In a further example, Modification Interface 130 comprises the functionality to access, create, delete, modify, and/or perform other manipulations on functions, methods, procedures, routines, subroutines, and/or other elements of Application Program 18. In a further example, Modification Interface 130 comprises the functionality to access, create, delete, modify, and/or perform other manipulations on source code, bytecode, compiled, interpreted, or otherwise translated code, machine code, and/or other code. In a further example, Modification Interface 130 comprises the functionality to access, create, delete, modify, and/or perform other manipulations on values, variables, parameters, and/or other data or information. Modification Interface 130 comprises any features, functionalities, and embodiments of Acquisition Interface 120, and vice versa. Modification Interface 130 also comprises other disclosed functionalities.

Modification Interface 130 can employ various techniques for modifying execution and/or functionality of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. In some aspects, some of the previously described techniques and/or tools can be utilized. Code instrumentation, for instance, may involve inserting additional code, overwriting or rewriting existing code, and/or branching to a separate segment of code in Application Program 18 as previously described. For example, instrumented code may include the following:

```
Object1.moveLeft(12);
modifyApplication();
```

In the above sample code, instrumented call to Modification Interface's 130 function (i.e. modifyApplication(), etc.) can be placed after a function (i.e. moveLeft(12), etc.) of Application Program 18. Similar call to an application modifying function can be placed after or before some or all functions/routines/subroutines, some or all lines of code, some or all statements, some or all instructions or instruction sets, some or all basic blocks, and/or some or all other code segments of Application Program 18. One or more application modifying function calls can be placed anywhere in Application Program's 18 code and can be executed at any points in Application Program's 18 execution. The application modifying function (i.e. modifyApplication(), etc.) may include Artificial Intelligence Unit 110-determined anticipatory Instruction Sets 526 that can modify execution and/or functionality of Application Program 18. In some embodiments, the previously described obtaining Application Program's 18 instruction sets, data, and/or other information as well as modifying execution and/or functionality of Application Program 18 can be implemented in a single function that performs both tasks (i.e. traceAndModifyApplication(), etc.).

In some embodiments, various computing systems and/or platforms may provide native tools for modifying execution and/or functionality of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. Independent vendors may provide tools with similar functionalities that can be utilized across different platforms. These tools enable a wide range of techniques or capabilities such as instrumentation, self-modifying code capabilities, dynamic code capabilities, branching, code rewriting, code overwriting, hot swapping, accessing and/or modifying objects or data structures, accessing and/or modifying functions/routines/subroutines, accessing and/or modifying variable or parameter values, accessing and/or modifying processor registers, accessing and/or modifying inputs and/or outputs, providing runtime memory access, and/or other capabilities. One of ordinary skill in art will understand that, while all possible variations of the techniques for modifying execution and/or functionality of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element are too voluminous to describe, these techniques are within the scope of this disclosure.

In one example, modifying execution and/or functionality of Application Program 18 can be implemented through utilizing metaprogramming techniques, which include applications that can self-modify or that can create, modify, and/or manipulate other applications. Self-modifying code, dynamic code, reflection, and/or other techniques can be used to facilitate metaprogramming. In some aspects, metaprogramming is facilitated through a programming language's ability to access and manipulate the internals of the runtime engine directly or via an API. In other aspects, metaprogramming is facilitated through dynamic execution of expressions (i.e. anticipatory Instruction Sets 526, etc.) that can be created and/or executed at runtime. In yet other aspects, metaprogramming is facilitated through application modification tools, which can perform modifications on an application regardless of whether the application's programming language enables metaprogramming capabilities. Some operating systems may protect an application loaded into memory by restricting access to the loaded application. This protection

mechanism can be circumvented by utilizing operating system's, processor's, and/or other low level features or commands to unprotect the loaded application. For example, a self-modifying application may modify the in-memory image of itself. To do so, the application can obtain the in-memory address of its code. The application may then change the operating system's or platform's protection on this memory range allowing it to modify the code (i.e. insert anticipatory Instruction Sets 526, etc.). In addition to a self-modifying application, one application can utilize similar technique to modify another application. Linux mprotect command or similar commends of other operating systems can be used to change protection (i.e. unprotect, etc.) for a region of memory, for example. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program 18 can be implemented through native capabilities of dynamic, interpreted, and/or scripting programming languages and/or platforms. Most of these languages and/or platforms can perform functionalities at runtime that static programming languages may perform during compilation. Dynamic, interpreted, and/or scripting languages provide native functionalities such as self-modification of code, dynamic code, extending the application, adding new code, extending objects and definitions, and/or other functionalities that can modify an application's execution and/or functionality at runtime. Examples of dynamic, interpreted, and/or scripting languages include Lisp, Perl, PHP, JavaScript, Ruby, Python, Smalltalk, Tcl, VBScript, and/or others. Similar functionalities can also be provided in languages such as Java, C, and/or others using reflection. Reflection includes the ability of an application to examine and modify the structure and behavior of the application at runtime. For example, JavaScript can modify its own code as it runs by utilizing Function object constructor as follows:

```
myFunc=new Function(arg1, arg2, argN, functionBody);
```

The sample code above causes a new function object to be created with the specified arguments and body. The body and/or arguments of the new function object may include new instruction sets (i.e. anticipatory Instruction Sets 526, etc.). The new function can be invoked as any other function in the original code. In another example, JavaScript can utilize eval method that accepts a string of JavaScript statements (i.e. anticipatory Instruction Sets 526, etc.) and execute them as if they were within the original code. An example of how eval method can be used to modify an application includes the following JavaScript code:

```
anticipatoryInstr = 'Object1.moveForward(32);';
if (anticipatoryInstr != "" && anticipatoryInstr != null)
{
    eval(anticipatoryInstr);
}
```

In the sample code above, Artificial Intelligence Unit 110 may generate anticipatory Instruction Set 526 (i.e. 'Object1.moveForward(32)' for moving an object forward 32 units, etc.) and save it in anticipatoryInstr variable, which eval method can then execute. Lisp is another example of dynamic, interpreted, and/or scripting language that includes similar capabilities as previously described JavaScript. For example, Lisp's compile command can create a function at runtime, eval command may parse and evaluate an expression at runtime, and exec command may execute a given instruction set (i.e. string, etc.) at runtime. In another example, dynamic as well as some non-dynamic languages may provide macros, which combine code introspection and/or eval capabilities. In some aspects, macros can access inner workings of the compiler, interpreter, virtual machine, runtime

environment/engine, and/or other components of the computing platform enabling the definition of language-like constructs and/or generation of a complete program or sections thereof. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program 18 can be implemented through dynamic code, dynamic class loading, reflection, and/or other native functionalities of a programming language or platform. In static applications or static programming, a class can be defined and/or loaded at compile time. Conversely, in dynamic applications or dynamic programming, a class can be loaded into a running environment at runtime. For example, Java Runtime Environment (JRE) may not require that all classes be loaded at compile time and class loading can occur when a class is first referenced at runtime. Dynamic class loading enables inclusion or injection of on-demand code and/or functionalities at runtime. System provided or custom class loaders may enable loading of classes into the running environment. Custom class loaders can be created to enable custom functionalities such as, for example, specifying a remote location from which a class can be loaded. In addition to dynamic loading of a pre-defined class, a class can also be created at runtime. In some aspects, a class source code can be created at runtime. A compiler such as javac, com.sun.tools.javac.Main, javax.tools, javax.tools.JavaCompiler, and/or other packages can then be utilized to compile the source code. Javac, com.sun.tools.javac.Main, javax.tools, javax.tools.JavaCompiler, and/or other packages may include an interface to invoke Java compiler from within a running application. A Java compiler may accept source code in a file, string, object (i.e. Java String, StringBuffer, CharSequence, etc.) and/or other source, and may generate Java bytecode (i.e. class file, etc.). Once compiled, a class loader can then load the compiled class into the running environment. In other aspects, a tool such as Javaassist (i.e. Java programming assistant) can be utilized to enable an application to create or modify a class at runtime. Javassist may include a Java library that provides functionalities to create and/or manipulate Java bytecode of an application as well as reflection capabilities. Javassist may provide source-level and bytecode-level APIs. Using the source-level API, a class can be created and/or modified using only source code, which Javassist may compile seamlessly on the fly. Javassist source-level API can therefore be used without knowledge of Java bytecode specification. Bytecode-level API enables creating and/or editing a class bytecode directly. In yet other aspects, similar functionalities to the aforementioned ones may be provided in tools such as Apache Commons BCEL (Byte Code Engineering Library), ObjectWeb ASM, CGLIB (Byte Code Generation Library), and/or others. Once a dynamic code or class is created and loaded, reflection in high-level programming languages such as Java and/or others can be used to manipulate or change the runtime behavior of an application. Examples of reflective programming languages and/or platforms include Java, JavaScript, Smalltalk, Lisp, Python, .NET Common Language Runtime (CLR), Tcl, Ruby, Perl, PHP, Scheme, PL/SQL, and/or others. Reflection can be used in an application to access, examine, modify, and/or manipulate a loaded class and/or its elements. Reflection in Java can be implemented by utilizing a reflection API such as java.lang.Reflect package. The reflection API provides functionalities such as, for example, loading or reloading a class, instantiating a new instance of a class, determining class and instance methods, invoking class and instance methods, accessing and manipulating a class, fields, methods and constructors, determining the modifiers for fields, methods, classes, and interfaces, and/or other functionalities. The above described dynamic code, dynamic class loading, reflection, and/or other functionalities are similarly provided in the .NET platform through its tools such as, for example, System.CodeDom.Compiler namespace, System.Reflection.Emit namespace, and/or other native or other .NET tools. Other platforms in addition

to Java and .NET may provide similar tools and/or functionalities. In some designs, dynamic code, dynamic class loading, reflection, and/or other functionalities can be used to facilitate modification of an application by inserting or injecting instruction sets (i.e. anticipatory Instruction Sets 526, etc.) into a running application. For example, an existing or dynamically created class comprising VSADO Unit 100 functionalities can be loaded into a running application through manual, automatic, or dynamic instrumentation. Once the class is created and loaded, an instance of VSADO Unit 100 class may be constructed. The instance of VSADO Unit 100 can then take or exert control of the application and/or implement alternate instruction sets (i.e. anticipatory Instruction Sets 526, etc.) at any point in the application's execution. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program 18 can be implemented through independent tools that can be utilized across different platforms. Such tools provide instrumentation and/or other capabilities on more than one platform or computing system and may facilitate application modification or insertion of instruction sets (i.e. anticipatory Instruction Sets 526, etc.). Examples of these tools include Pin, DynamoRIO, DynInst, Kprobes, KernInst, OpenPAT, DTrace, SystemTap, and/or others. In some aspects, Pin and/or any of its elements, methods, and/or techniques can be utilized for dynamic instrumentation. Pin can perform instrumentation by taking control of an application after it loads into memory. Pin may insert itself into the address space of an executing application enabling it to take control. Pin JIT compiler can then compile and implement alternate code (i.e. anticipatory Instruction Sets 526, etc.). Pin provides an extensive API for instrumentation at several abstraction levels. Pin supports two modes of instrumentation, JIT mode and probe mode. JIT mode uses a just-in-time compiler to insert instrumentation and recompile program code while probe mode uses code trampolines for instrumentation. Pin was designed for architecture and operating system independence. In other aspects, KernInst and/or any of its elements, methods, and/or techniques can be utilized for dynamic instrumentation. KernInst includes an instrumentation framework designed for dynamically inserting code into a running kernel of an operating system. KernInst implements probe-based dynamic instrumentation where code can be inserted, changed, and/or removed at will. Kerninst API enables client tools to construct their own tools for dynamic kernel instrumentation to suit variety of purposes such as insertion of alternate instruction sets (i.e. anticipatory Instruction Sets 526, etc.). Client tools can communicate with KernInst over a network (i.e. internet, wireless network, LAW, WAN, etc). Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program 18 can be implemented through utilizing operating system's native tools or capabilities such as Unix ptrace command. Ptrace includes a system call that may enable one process to control another allowing the controller to inspect and manipulate the internal state of its target. Ptrace can be used to modify a running application such as modifying an application with alternate instruction sets (i.e. anticipatory Instruction Sets 526, etc.). By attaching to an application using the ptrace call, the controlling application can gain extensive control over the operation of its target. This may include manipulation of its instruction sets, execution path, file descriptors, memory, registers, and/or other components. Ptrace can single-step through the target's code, observe and intercept system calls and their results, manipulate the target's signal handlers, receive and send signals on the target's behalf, and/or perform other operations within the target application. Ptrace's ability to write into the target application's memory space enables

the controller to modify the running code of the target application. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program 18 can be implemented through utilizing just-in-time (JIT) compiling. JIT compilation (also known as dynamic translation, dynamic compilation, etc.) includes compilation performed during an application's execution (i.e. runtime, etc.). A code can be compiled when it is about to be executed, and it may be cached and reused later without the need for additional compilation. In some aspects, a JIT compiler can convert source code or byte code into machine code. In other aspects, a JIT compiler can convert source code into byte code. JIT compiling may be performed directly in memory. For example, JIT compiler can output machine code directly into memory and immediately execute it. Platforms such as Java, .NET, and/or others may implement JIT compilation as their native functionality. Platform independent tools for custom system design may include JIT compilation functionalities as well. In some aspects, JIT compilation includes redirecting application's execution to a JIT compiler from a specific entry point. For example, Pin can insert its JIT compiler into the address space of an application. Once execution is redirected to it, JIT compiler may receive alternate instruction sets (i.e. anticipatory Instruction Sets 526, etc.) immediately before their compilation. The JIT compiled instruction sets can be stored in memory or another repository from where they may be retrieved and executed. Alternatively, for example, JIT compiler can create a copy of the original application code or a segment thereof, and insert alternate code (i.e. anticipatory Instruction Sets 526, etc.) before compiling the modified code copy. In some aspects, JIT compiler may include a specialized memory such as fast cache memory dedicated to JIT compiler functionalities from which the modified code can be fetched rapidly. JIT compilation and/or any compilation in general may include compilation, interpretation, or other translation into machine code, bytecode, and/or other formats or types of code. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program 18 can be implemented through dynamic recompilation. Dynamic recompilation includes recompiling an application or part thereof during execution. An application can be modified with alternate features or instruction sets that may take effect after recompilation. Dynamic recompilation may be practical in various types of applications including object oriented, event driven, forms based, and/or other applications. In a typical windows-based application, most of the action after initial startup occurs in response to user or system events such as moving the mouse, selecting a menu option, typing text, running a scheduled task, making a network connection, and/or other events when an event handler is called to perform an operation appropriate for the event. Generally, when no events are being generated, the application is idle. For example, when an event occurs and an appropriate event handler is called, instrumentation can be implemented in the application's source code to insert alternate instruction sets (i.e. anticipatory Instruction Sets 526, etc.) at which point the modified source code can be recompiled and/or executed. In some aspects, the state of the application can be saved before recompiling its modified source code so that the application may continue from its prior state. Saving the application's state can be achieved by saving its variables, data structures, objects, location of its current instruction, and/or other necessary information in environmental variables, memory, or other repositories where they can be accessed once the application is recompiled. In other aspects, application's variables, data structures, objects, address of its current instruction, and/or other necessary information can be saved in a repository such as file, database, or other repository accessible to the application after

recompilation of its source code. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program 18 can be implemented through modifying or redirecting Application Program's 18 execution path. Generally, an application can be loaded into memory and the flow of execution proceeds from one instruction set to the next until the end of the application. An application may include a branching mechanism that can be driven by keyboard or other input devices, system events, and/or other computing system components or events that may impact the execution path. The execution path can also be altered by an external application through acquiring control of execution and/or redirecting execution to a function, routine/subroutine, or an alternate code segment at any point in the application's execution. A branch, jump, or other mechanism can be utilized to implement the redirected execution. For example, a jump instruction can be inserted at a specific point in an application's execution to redirect execution to an alternate code segment. A jump instruction set may include, for example, an unconditional branch, which always results in branching, or a conditional branch, which may or may not result in branching depending on a condition. When executing an application, a computer may fetch and execute instruction sets in sequence until it encounters a branch instruction set. If the instruction set is an unconditional branch, or it is conditional and the condition is satisfied, the computer may fetch its next instruction set from a different instruction set sequence or code segment as specified by the branch instruction set. After the execution of the alternate code segment, control may be redirected back to the original jump point or to another point in the application. For example, modifying an application can be implemented by redirecting execution of an application to alternate instruction sets (i.e. anticipatory Instruction Sets 526, etc.). Alternate instruction sets can be pre-compiled, pre-interpreted, or otherwise pre-translated and ready for execution. Alternate instruction sets can also be JIT compiled, JIT interpreted, or otherwise JIT translated before execution. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program 18 can be implemented through assembly language. Assembly language instructions may be directly related with the architecture's machine instructions as previously described. Assembly language can, therefore, be a powerful tool for implementing direct hardware (i.e. processor registers, memory, etc.) access and manipulations as well as access and manipulations of specialized processor features or instructions. Assembly language can also be a powerful tool for implementing low-level embedded systems, real-time systems, interrupt handlers, self or dynamically modifying code, and/or other applications. Specifically, for instance, self or dynamically modifying code that can be used to facilitate modifying of an application can be seamlessly implemented using assembly language. For example, using assembly language, instruction sets can be dynamically created and loaded into memory similar to the ones that a compiler may generate. Furthermore, using assembly language, memory space of a loaded application can be accessed to modify (including rewrite, overwrite, etc.) original instruction sets or to insert jumps or branches to alternate code elsewhere in memory. Some operating systems may implement protection from changes to applications loaded into memory. Operating system's, processor's, or other low level features or commands can be used to unprotect the protected locations in memory before the change as previously described. Alternatively, a pointer that may reside in a memory location where it could be readily altered can be utilized where the pointer may reference alternate code. In one example, assembly language can be utilized to write alternate code (i.e.

anticipatory Instruction Sets 526, etc.) into a location in memory outside a running application's memory space. Assembly language can then be utilized to redirect the application's execution to the alternate code by inserting a jump or branch into the application's in-memory code, by redirecting program counter, or by other technique. In another example, assembly language can be utilized to overwrite or rewrite the entire or part of an application's in-memory code with alternate code. In some aspects, high-level programming languages can call an external assembly language program to facilitate application modification as previously described. In yet other aspects, relatively low-level programming languages such as C may allow embedding assembly language directly in their source code such as, for example, using asm keyword of C. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program 18 can be implemented through binary rewriting. Binary rewriting tools and/or techniques may modify an application's executable. In some aspects, modification can be minor such as in the case of optimization where the original executable's functionality is kept. In other aspects, modification may change the application's functionality such as by inserting alternate code (i.e. anticipatory Instruction Sets 526, etc.). Examples of binary rewriting tools include SecondWrite, ATOM, DynamoRIO, Purify, Pin, EEL, DynInst, PLTO, and/or others. Binary rewriting may include disassembly, analysis, and/or modification of target application. Since binary rewriting works directly on machine code executable, it is independent of source language, compiler, virtual machine (if one is utilized), and/or other higher level abstraction layers. Also, binary rewriting tools can perform application modifications without access to original source code. Binary rewriting tools include static rewriters, dynamic rewriters, minimally-invasive rewriters, and/or others. Static binary rewriters can modify an executable when the executable is not in use (i.e. not running). The rewritten executable may then be executed including any new or modified functionality. Dynamic binary rewriters can modify an executable during its execution, thereby enabling modification of an application's functionality at runtime. In some aspects, dynamic rewriters can be used for instrumentation or selective modifications such as insertion of alternate code (i.e. anticipatory Instruction Sets 526, etc.), and/or for other runtime transformations or modifications. For example, some dynamic rewriters can be configured to intercept an application's execution at indirect control transfers and insert instrumentation or other application modifying code. Minimally-invasive rewriters may keep the original machine code to the greatest extent possible. They support limited modifications such as insertion of jumps into and out of instrumented code. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

Referring to Fig. 29, in a further example, modifying execution and/or functionality of Processor 11 can be implemented through modification of processor registers, memory, or other computing system components. In some aspects, modifying execution and/or functionality of Processor 11 can be implemented by redirecting Processor's 11 execution to alternate instruction sets (i.e. anticipatory Instruction Sets 526, etc.). In one example, Program Counter 211 may hold or point to a memory address of the next instruction set that will be executed by Processor 11. Artificial Intelligence Unit 110 may generate anticipatory Instruction Sets 526 and store them in Memory 12 as previously described. Modification Interface 130 may then change Program Counter 211 to point to the location in Memory 12 where anticipatory Instruction Sets 526 are stored. The anticipatory Instruction Sets 526 can then be fetched from the location in Memory 12 pointed to by the modified Program Counter 211 and loaded into Instruction Register 212 for decoding and execution. Once anticipatory Instruction Sets 526 are executed, Modification

Interface 130 may change Program Counter 211 to point to the last instruction set before the redirection or to any other instruction set. In other aspects, anticipatory Instruction Sets 526 can be loaded directly into Instruction Register 212. As previously described, examples of other processor or computing system components that can be used during an instruction cycle include memory address register (MAR), memory data register (MDR), data registers, address registers, general purpose registers (GPRs), conditional registers, floating point registers (FPRs), constant registers, special purpose registers, machine-specific registers, Register Array 214, Arithmetic Logic Unit 215, control unit, and/or other circuits or components. Any of the aforementioned processor registers, memory, or other computing system components can be accessed and/or modified to facilitate the disclosed functionalities. In some embodiments, processor interrupt may be issued to facilitate such access and/or modification. In some designs, modifying execution and/or functionality of Processor 11 can be implemented in a program, combination of programs and hardware, or purely hardware system. Dedicated hardware may be built to perform modifying execution and/or functionality of Processor 11 with marginal or no impact to computing overhead. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

Referring to Figs. 30A-30B, in a further example, modifying execution and/or functionality of Logic Circuit 250 can be implemented through modification of inputs and/or outputs of Logic Circuit 250. While Processor 11 includes any type of logic circuit, Logic Circuit 250 is described separately herein to offer additional detail on its functioning. Logic Circuit 250 comprises the functionality for performing logic operations using the circuit's inputs and producing outputs based on the logic operations performed as previously described. In one example, Logic Circuit 250 may perform some logic operations using four input values and produce two output values. Modifying execution and/or functionality of Logic Circuit 250 can be implemented by replacing its input values with anticipatory input values (i.e. anticipatory Instruction Sets 526, etc.). Artificial Intelligence Unit 110 may generate anticipatory input values as previously described. Modification Interface 130 can then transmit the anticipatory input values to Logic Circuit 250 through the four hardwired connections as shown in Fig. 30A. Modification Interface 130 may use Switches 251 to prevent delivery of any input values that may be sent to Logic Circuit 250 from its usual input source. As such, VSADO Unit 100 may cause Logic Circuit 250 to perform its logic operations using the four anticipatory input values, thereby implementing autonomous Device 98 operation. In another example, Logic Circuit 250 may perform some logic operations using four input values and produce two output values. Modifying execution and/or functionality of Logic Circuit 250 can be implemented by replacing its output values with anticipatory output values (i.e. anticipatory Instruction Sets 526, etc.). Artificial Intelligence Unit 110 may generate anticipatory output values (i.e. anticipatory Instruction Sets 526, etc.) as previously described. Modification Interface 130 can then transmit the anticipatory output values through the two hardwired connections as shown in Fig. 30B. Modification Interface 130 may use Switches 251 to prevent delivery of any output values that may be sent by Logic Circuit 250. As such, VSADO Unit 100 may bypass Logic Circuit 250 and transmit the two anticipatory output values to downstream elements, thereby implementing autonomous Device 98 operation. In a further example, instead of or in addition to modifying input and/or output values of Logic Circuit 250, the execution and/or functionality of Logic Circuit 250 may be modified by modifying values or signals in one or more Logic Circuit's 250 internal components such as registers, memories, buses, and/or others (i.e. similar to the previously described modifying of Processor 11 components, etc.). In some designs, modifying execution and/or functionality of Logic Circuit 250 can be implemented in a program, combination of programs and hardware, or purely hardware system. Dedicated hardware

may be built to perform modifying execution and/or functionality of Logic Circuit 250 with marginal or no impact to computing overhead. Any of the elements and/or techniques for modifying execution and/or functionality of Logic Circuit 250 can similarly be implemented with Processor 11 and/or other processing elements.

In some embodiments, VSADO Unit 100 may directly modify the functionality of an actuator (previously described, not shown). For example, Logic Circuit 250 or other processing element may control an actuator that enables Device 98 to perform mechanical, physical, and/or other operations. An actuator may receive one or more input values or control signals from Logic Circuit 250 or other processing element directing the actuator to perform specific operations. Modifying functionality of an actuator can be implemented by replacing its input values with anticipatory input values (i.e. anticipatory Instruction Sets 526, etc.) as previously described with respect to replacing input values of Logic Circuit 250. Specifically, for instance, Artificial Intelligence Unit 110 may generate anticipatory input values as previously described. Modification Interface 130 can then transmit the anticipatory input values to the actuator. Modification Interface 130 may use Switches 251 to prevent delivery of any input values that may be sent to the actuator from its usual input source. As such, VSADO Unit 100 may cause the actuator to perform its operations using the anticipatory input values, thereby implementing autonomous Device 98 operation.

One of ordinary skill in art will recognize that Figs. 30A-30B depict one of many implementations of Logic Circuit 250 and that any number of input and/or output values can be utilized in alternate implementations. One of ordinary skill in art will also recognize that Logic Circuit 250 may include any number and/or combination of logic components to implement any logic operations.

Other additional techniques or elements can be utilized as needed for modifying execution and/or functionality of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing elements, or some of the disclosed techniques or elements can be excluded, or a combination thereof can be utilized in alternate embodiments.

Referring to Fig. 31, the illustration shows an embodiment of a method 6100 for learning and/or using visual surrounding for autonomous device operation. The method can be used on a computing device or system to enable learning of a device’s operation in various visual surroundings and enable autonomous device operation in similar visual surroundings. Method 6100 may include any action or operation of any of the disclosed methods such as method 6200, 6300, 6400, 6500, 6600, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method 6100.

At step 6105, a first digital picture is received. A digital picture (i.e. Digital Picture 525, etc.) may include a depiction of a device’s (i.e. Device’s 98, etc.) visual surrounding. A digital picture may include a depiction of a remote device’s (i.e. Remote Device’s 97, etc.) visual surrounding. In some embodiments, a digital picture may include a collection of color encoded pixels or dots. A digital picture comprises any type or form of digital picture such as JPEG, GIF, TIFF, PNG, PDF, and/or other digitally encoded picture. In other embodiments, a stream of digital pictures (i.e. motion picture, video, etc.) may include one or more digital pictures. A stream of digital pictures comprises any type or form of digital motion picture such as MPEG, AVI, FLV, MOV, RM, SWF, WMV, DivX, and/or other digitally encoded motion picture. In some aspects, a digital picture may include or be substituted with a stream of digital pictures, and vice versa. Therefore, the terms digital picture and stream of digital pictures may be used interchangeably herein depending on context. One or more digital pictures can be captured by a picture capturing

apparatus (i.e. Picture Capturing Apparatus 90, etc.) such as a still or motion picture camera, or other picture capturing apparatus. In some aspects, a picture capturing apparatus may be part of a device whose visual surrounding is being used for VSADO functionalities. In other aspects, a picture capturing apparatus may be part of a remote device, accessible via a network, whose visual surrounding is being used for VSADO functionalities. Picture capturing apparatus may be provided in any other device, system, process, or configuration. In some embodiments, capturing and/or receiving may be responsive to a triggering object, action, event, time, and/or other stimulus. Receiving comprises any action or operation by or for a Picture Capturing Apparatus 90, Digital Picture 525, and/or other disclosed elements.

At step 6110, one or more instruction sets for operating a device are received. In some embodiments, an instruction set (i.e. Instruction Set 526, etc.) may be used or executed by a processor (i.e. Processor 11, etc.) for operating a device (i.e. Device 98, etc.). In other embodiments, an instruction set may be part of an application program (i.e. Application Program 18, etc.) for operating a device. The application can run or execute on one or more processors or other processing elements. In further embodiments, an instruction set may be used, executed, or produced by a logic circuit (i.e. Logic Circuit 250, etc.) for operating a device. For example, such instruction set may be or include one or more inputs into or outputs from a logic circuit. In further embodiments, an instruction set may be used by an actuator for operating a device. For example, such instruction set may be or include one or more inputs into an actuator. Operating a device includes performing any operations on or with the device. An instruction set may temporally correspond to a digital picture. In some aspects, an instruction set that temporally corresponds to a digital picture may include an instruction set used or executed at the time of receiving or capturing the digital picture. In other aspects, an instruction set that temporally corresponds to a digital picture may include an instruction set used or executed within a certain time period before and/or after receiving or capturing the digital picture. Any time period may be utilized. In further aspects, an instruction set that temporally corresponds to a digital picture may include an instruction set used or executed from the time of capturing of the digital picture to the time of capturing of a next digital picture. In further aspects, an instruction set that temporally corresponds to a digital picture may include an instruction set used or executed from the time of capturing of a preceding digital picture to the time of capturing of the digital picture. Any other temporal relationship or correspondence between digital pictures and correlated instruction sets can be implemented. In general, an instruction set that temporally corresponds to a digital picture enables structuring knowledge of a device's operation at or around the time of the receiving or capturing the digital picture. Such functionality enables spontaneous or seamless learning of a device's operation in various visual surroundings as user operates the device in real life situations. In some designs, an instruction set can be received from a processor, application program, logic circuit, and/or other processing element as the instruction set is being used or executed. In other aspects, an instruction set can be received from a processor, application program, logic circuit, and/or other processing element after the instruction set is used or executed. In further aspects, an instruction set can be received from a processor, application program, logic circuit, and/or other processing element before the instruction set has been used or executed. An instruction set can be received from a running processor, running application program, running logic circuit, and/or other running processing element. As such, an instruction set can be received at runtime. In other designs, an instruction set can be received from an actuator. In some embodiments, an instruction set may include one or more commands, keywords, symbols (i.e. parentheses, brackets, commas, semicolons, etc.), instructions, operators (i.e. =, <, >, etc.), variables, values, objects (i.e. file

handle, network connection, Object1, etc.), data structures (i.e. table, database, user defined data structure, etc.), functions (i.e. Function1(), FIRST(), MIN(), SQRT(), etc.), parameters, states, signals, inputs, outputs, references thereto, and/or other components for performing an operation. In other embodiments, an instruction set may include source code, bytecode, intermediate code, compiled, interpreted, or otherwise translated code, runtime code, assembly code, machine code, and/or any other computer code. In further embodiments, an instruction set can be compiled, interpreted or otherwise translated into machine code or any intermediate code (i.e. bytecode, assembly code, etc.). In further embodiments, an instruction set may include one or more inputs into and/or outputs from a logic circuit. In further embodiments, an instruction set may include one or more inputs into an actuator. In some aspects, an instruction set can be received from memory (i.e. Memory 12, etc.), hard drive, or any other storage element or repository. In other aspects, an instruction set can be received over a network such as Internet, local area network, wireless network, and/or other network. In further aspects, an instruction set can be received by an interface (i.e. Acquisition Interface 120, etc.) configured to obtain instruction sets from a processor, application program, logic circuit, actuator, and/or other element. In general, an instruction set can be received by any element of the system. In some embodiments, receiving may be responsive to a triggering object, action, event, time, and/or other stimulus. Receiving comprises any action or operation by or for an Acquisition Interface 120, Instruction Set 526, and/or other disclosed elements.

At step 6115, the first digital picture is correlated with the one or more instruction sets for operating the device. In some aspects, individual digital pictures can be correlated with one or more instruction sets. In other aspects, streams of digital pictures can be correlated with one or more instruction sets. In further aspects, individual digital pictures or streams of digital pictures can be correlated with temporally corresponding instruction sets as previously described. In further aspects, a digital picture or stream of digital pictures may not be correlated with any instruction sets. Correlating may include structuring or generating a knowledge cell (i.e. Knowledge Cell 800, etc.) and storing one or more digital pictures correlated with any instruction sets into the knowledge cell. Therefore, knowledge cell may include any data structure or arrangement that can facilitate such storing. A knowledge cell includes a unit of knowledge of how a device operated in a visual surrounding. In some designs, extra information (i.e. Extra Info 527, etc.) may optionally be used to facilitate enhanced comparisons or decision making in autonomous device operation where applicable. Therefore, any digital picture, instruction set, and/or other element may include or be correlated with extra information. Extra information may include any information useful in comparisons or decision making performed in autonomous device operation. Examples of extra information include time information, location information, computed information, observed information, sensory information, contextual information, and/or other information. In some embodiments, correlating may be responsive to a triggering object, action, event, time, and/or other stimulus. Correlating may be omitted where learning of a device's operations in visual surroundings is not implemented. Correlating comprises any action or operation by or for a Knowledge Structuring Unit 520, Knowledge Cell 800, and/or other disclosed elements.

At step 6120, the first digital picture correlated with the one or more instruction sets for operating the device is stored. A digital picture correlated with one or more instruction sets may be part of a stored plurality of digital pictures correlated with one or more instruction sets. Digital pictures correlated with any instruction sets can be stored in a memory unit or other repository. The previously described knowledge cells comprising digital pictures correlated with any instruction sets can be used in/as neurons, nodes, vertices, or other elements in any of the data

structures or arrangements (i.e. neural networks, graphs, sequences, collection of knowledge cells, etc.) used for storing the knowledge of a device's operation in visual surroundings. Knowledge cells may be connected, interrelated, or interlinked into knowledge structures using statistical, artificial intelligence, machine learning, and/or other models or techniques. Such interconnected or interrelated knowledge cells can be used for enabling autonomous device operation. The interconnected or interrelated knowledge cells may be stored or organized into a knowledgebase (i.e. Knowledgebase 530, etc.). In some embodiments, knowledgebase may be or include a neural network (i.e. Neural Network 530a, etc.). In other embodiments, knowledgebase may be or include a graph (i.e. Graph 530b, etc.). In further embodiments, knowledgebase may be or include a collection of sequences (i.e. Collection of Sequences 530c, etc.). In further embodiments, knowledgebase may be or include a sequence (i.e. Sequence 533, etc.). In further embodiments, knowledgebase may be or include a collection of knowledge cells (i.e. Collection of Knowledge Cells 530d, etc.). In general, knowledgebase may be or include any data structure or arrangement, and/or repository capable of storing the knowledge of a device's operation in various visual surroundings. Knowledgebase may also include or be substituted with various artificial intelligence methods, systems, and/or models for knowledge structuring, storing, and/or representation such as deep learning, supervised learning, unsupervised learning, neural networks (i.e. convolutional neural network, recurrent neural network, deep neural network, etc.), search-based, logic and/or fuzzy logic-based, optimization-based, tree/graph/other data structure-based, hierarchical, symbolic and/or sub-symbolic, evolutionary, genetic, multi-agent, deterministic, probabilistic, statistical, and/or other methods, systems, and/or models. Storing may be omitted where learning of a device's operations in visual surroundings is not implemented. Storing comprises any action or operation by or for a Knowledgebase 530, Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, Knowledge Cell 800, Node 852, Layer 854, Connection 853, Similarity Comparison 125, and/or other disclosed elements.

At step 6125, a new digital picture is received. Step 6125 may include any action or operation described in Step 6105 as applicable.

At step 6130, the new digital picture is compared with the first digital picture. Comparing one digital picture with another digital picture may include comparing at least a portion of one digital picture with at least a portion of the other digital picture. In some embodiments, digital pictures may be compared individually. In some aspects, comparing of individual pictures may include comparing one or more regions of one picture with one or more regions of another picture. In other aspects, comparing of individual pictures may include comparing one or more features of one picture with one or more features of another picture. In further aspects, comparing of individual pictures may include comparing pixels of one picture with pixels of another picture. In other aspects, comparing of individual pictures may include recognizing a person or object in one digital picture and recognizing a person or object in another digital picture, and comparing the person or object from the one digital picture with the person or object from the other digital picture. Comparing may also include other aspects or properties of digital pictures or pixels examples of which comprise color adjustment, size adjustment, content manipulation, transparency (i.e. alpha channel, etc.), use of a mask, and/or others. In other embodiments, digital pictures may be compared collectively as part of streams of digital pictures (i.e. motion pictures, videos, etc.). In some aspects, collective comparing may include comparing one or more digital pictures of one stream of digital pictures with one or more digital pictures of another stream of digital pictures. In some aspects, Dynamic Time Warping (DTW) and/or other techniques can be

utilized for comparison and/or aligning temporal sequences (i.e. streams of digital pictures, etc.) that may vary in time or speed. Any combination of the aforementioned and/or other elements or techniques can be utilized in alternate embodiments of the comparing. Comparing may be omitted where anticipating of a device's operation in visual surroundings is not implemented. Comparing comprises any action or operation by or for a Decision-making Unit 540, Similarity Comparison 125, and/or other disclosed elements.

At step 6135, a determination is made that there is at least a partial match between the new digital picture and the first digital picture. In some embodiments, determining at least a partial match between individually compared digital pictures includes determining that similarity between one or more portions of one digital picture and one or more portions of another digital picture exceeds a similarity threshold. In other embodiments, determining at least a partial match between individually compared digital pictures includes determining at least a partial match between one or more portions of one digital picture and one or more portions of another digital picture. A portion of a digital picture may include a region, a feature, a pixel, or other portion. In further embodiments, determining at least a partial match between individually compared digital pictures includes determining that the number or percentage of matching or substantially matching regions of the compared pictures exceeds a threshold number (i.e. 1, 2, 5, 11, 39, etc.) or threshold percentage (i.e. 38%, 63%, 77%, 84%, 98%, etc.). In some aspects, the type of regions, the importance of regions, and/or other elements or techniques relating to regions can be utilized for determining similarity using regions. In further aspects, some of the regions can be omitted in determining similarity using regions. In further aspects, similarity determination can focus on regions of interest from the compared pictures. In further aspects, detection or recognition of persons or objects in regions of the compared pictures can be utilized for determining similarity. Where a reference to a region is used herein it should be understood that a portion of a region or a collection of regions can be used instead of or in addition to the region. In further embodiments, determining at least a partial match between individually compared digital pictures includes determining that the number or percentage of matching or substantially matching features of the compared pictures exceeds a threshold number (i.e. 3, 22, 47, 93, 128, 431, etc.) or a threshold percentage (i.e. 49%, 53%, 68%, 72%, 95%, etc.). In some aspects, the type of features, the importance of features, and/or other elements or techniques relating to features can be utilized for determining similarity using features. In further aspects, some of the features can be omitted in determining similarity using features. In further aspects, similarity determination can focus on features in certain regions of interest from the compared pictures. In further aspects, detection or recognition of persons or objects using features in the compared pictures can be utilized for determining similarity. Where a reference to a feature is used herein it should be understood that a portion of a feature or a collection of features can be used instead of or in addition to the feature. In further embodiments, determining at least a partial match between individually compared digital pictures may include determining that the number or percentage of matching or substantially matching pixels of the compared pictures exceeds a threshold number (i.e. 449, 2219, 92229, 442990, 1000028, etc.) or a threshold percentage (i.e. 39%, 45%, 58%, 72%, 92%, etc.). In some aspects, some of the pixels can be omitted in determining similarity using pixels. In further aspects, similarity determination can focus on pixels in certain regions of interest from the compared pictures. Where a reference to a pixel is used herein it should be understood that a collection of pixels can be used instead of or in addition to the pixel. In further embodiments, determining at least a partial match between individually compared digital pictures may include determining substantial similarity between at least a portion of one digital picture and at least a portion of another digital picture. In some aspects, substantial

similarity of individually compared digital pictures can be achieved when a similarity between at least a portion of one digital picture and at least a portion of another digital picture exceeds a similarity threshold. In other aspects, substantial similarity of individually compared digital pictures can be achieved when the number or percentage of matching or substantially matching regions of the compared pictures exceeds a threshold number (i.e. 3, 22, 47, 93, 128, 431, etc.) or a threshold percentage (i.e. 49%, 53%, 68%, 72%, 95%, etc.). In further aspects, substantial similarity of individually compared digital pictures can be achieved when the number or percentage of matching or substantially matching features of the compared pictures exceeds a threshold number (i.e. 1, 2, 5, 11, 39, etc.) or threshold percentage (i.e. 38%, 63%, 77%, 84%, 98%, etc.). In further aspects, substantial similarity of individually compared digital pictures can be achieved when the number or percentage of matching or substantially matching pixels of the compared pictures exceeds a threshold number (i.e. 449, 2219, 92229, 442990, 1000028, etc.) or a threshold percentage (i.e. 39%, 45%, 58%, 72%, 92%, etc.). In some designs, substantial similarity of individually compared digital pictures can be achieved taking into account objects or persons detected within the compared digital pictures. For example, substantial similarity can be achieved if same or similar objects or persons are detected in the compared pictures. In some embodiments, determining at least a partial match between collectively compared digital pictures (i.e. streams of digital pictures [i.e. motion pictures, videos, etc.], etc.) may include determining that the number or percentage of matching or substantially matching digital pictures of the compared streams of digital pictures exceeds a threshold number (i.e. 28, 74, 283, 322, 995, 874, etc.) or a threshold percentage (i.e. 29%, 33%, 58%, 72%, 99%, etc.). In some aspects, Dynamic Time Warping (DTW) and/or other techniques for aligning temporal sequences (i.e. streams of digital pictures, etc.) that may vary in time or speed can be utilized in determining similarity of collectively compared digital pictures or streams digital pictures. In other aspects, the order of digital pictures, the importance of digital pictures, and/or other elements or techniques relating to digital pictures can be utilized for determining similarity of collectively compared digital pictures or streams digital pictures. In further aspects, some of the digital pictures can be omitted in determining similarity of collectively compared digital pictures or streams digital pictures. In some designs, a threshold for a number or percentage similarity can be used to determine a match or substantial match between any of the aforementioned elements. Any combination of the aforementioned and/or other elements or techniques can be utilized in alternate embodiments. Determining may be omitted where anticipating of a device's operation in visual surroundings is not implemented. Determining comprises any action or operation by or for a Decision-making Unit 540, Similarity Comparison 125, and/or other disclosed elements.

At step 6140, the one or more instruction sets for operating the device correlated with the first digital picture are executed. The executing may be performed in response to the aforementioned determining. The executing may be caused by VSADO Unit 100, Artificial Intelligence Unit 110, and/or other disclosed elements. An instruction set may be executed by a processor (i.e. Processor 11, etc.), application program (i.e. Application Program 18, etc.), logic circuit (i.e. Logic Circuit 250, etc.), and/or other processing element. An instruction set may be executed or acted upon by an actuator. Executing may include executing one or more alternate instruction sets instead of or prior to an instruction set that would have been executed in a regular course of execution. In some aspects, alternate instruction sets comprise one or more instruction sets for operating a device correlated with one or more digital pictures. In some embodiments, executing may include modifying a register or other element of a processor with one or more alternate instruction sets. Executing may also include redirecting a processor to one or more

alternate instruction sets. In other embodiments, processor may be or comprises a logic circuit. Executing may include modifying an element of a logic circuit with one or more alternate instruction sets, redirecting the logic circuit to one or more alternate instruction sets, replacing the inputs into the logic circuit with one or more alternate inputs or instruction sets, and/or replacing the outputs from the logic circuit with one or more alternate outputs or instruction sets. In further embodiments, a processor may include an application including instruction sets for operating a device, the application running on the processor. In some aspects, executing includes executing one or more alternate instruction sets as part of the application. In other aspects, executing includes modifying the application. In further aspects, executing includes redirecting the application to one or more alternate instruction sets. In further aspects, executing includes modifying one or more instruction sets of the application. In further aspects, executing includes modifying the application's source code, bytecode, intermediate code, compiled code, interpreted code, translated code, runtime code, assembly code, machine code, or other code. In further aspects, executing includes modifying memory, processor register, storage, repository or other element where the application's instruction sets are stored or used. In further aspects, executing includes modifying instruction sets used for operating an object of the application. In further aspects, executing includes modifying an element of a processor, an element of a device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input used in running the application. In further aspects, executing includes modifying the application at source code write time, compile time, interpretation time, translation time, linking time, loading time, runtime, or other time. In further aspects, executing includes modifying one or more of the application's lines of code, statements, instructions, functions, routines, subroutines, basic blocks, or other code segments. In further aspects, executing includes a manual, automatic, dynamic, just in time (JIT), or other instrumentation of the application. In further aspects, executing includes utilizing one or more of a .NET tool, .NET application programming interface (API), Java tool, Java API, operating system tool, independent tool or other tool for modifying the application. In further aspects, executing includes utilizing a dynamic, interpreted, scripting or other programming language. In further aspects, executing includes utilizing dynamic code, dynamic class loading, or reflection. In further aspects, executing includes utilizing assembly language. In further aspects, executing includes utilizing metaprogramming, self-modifying code, or an application modification tool. In further aspects, executing includes utilizing just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further aspects, executing includes utilizing dynamic expression creation, dynamic expression execution, dynamic function creation, or dynamic function execution. In further aspects, executing includes adding or inserting additional code into the application's code. In further aspects, executing includes modifying, removing, rewriting, or overwriting the application's code. In further aspects, executing includes branching, redirecting, extending, or hot swapping the application's code. Branching or redirecting an application's code may include inserting a branch, jump, or other means for redirecting the application's execution. Executing comprises any action or operation by or for a Processor 11, Application Program 18, Logic Circuit 250, Modification Interface 130, and/or other disclosed elements.

At step 6145, one or more operations defined by the one or more instruction sets for operating the device correlated with the first digital picture are performed by the device. The one or more operations may be performed in response to the aforementioned executing. An operation includes any operation that can be performed by, with, or on the device. An operation includes any operation that can be performed by, with, or on an actuator. In one example, an operation includes any operation (i.e. moving, maneuvering, collecting, unloading, lifting, screwing,

gripping, etc.) with or by a computing enabled machine (i.e. Computing Enabled Machine 98a, etc.). In a further example, an operation includes any operation with or by a fixture (i.e. Fixture 98b, etc.). In a further example, an operation includes any operation (i.e. setting, starting, stopping, etc.) on or by a control device (i.e. Control Device 98c, etc.). In one example, an operation includes any operation on a smartphone (i.e. Smartphone 98d, etc.) or other mobile computer. In a further example, an operation includes any operation on or by a computer or computing enabled device. In a further example, an operation includes any motion or operation on or by an actuator. One of ordinary skill in art will recognize that, while all possible variations of operations on a device are too voluminous to list and limited only by the device's design and/or user's utilization, other operations are within the scope of this disclosure in various implementations.

Referring to Fig. 32, the illustration shows an embodiment of a method 6200 for learning and/or using visual surrounding for autonomous device operation. The method can be used on a computing device or system to enable learning of a device's operation in various visual surroundings and enable autonomous device operation in similar visual surroundings. Method 6200 may include any action or operation of any of the disclosed methods such as method 6100, 6300, 6400, 6500, 6600, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method 6200.

At step 6205, a first digital picture is received. Step 6205 may include any action or operation described in Step 6105 of method 6100 as applicable.

At step 6210, one or more instruction sets for operating a device are received. Step 6210 may include any action or operation described in Step 6110 of method 6100 as applicable.

At step 6215, the first digital picture correlated with the one or more instruction sets for operating the device are learned. Step 6215 may include any action or operation described in Step 6115 and/or Step 6120 of method 6100 as applicable.

At step 6220, a new digital picture is received. Step 6220 may include any action or operation described in Step 6125 of method 6100 as applicable.

At step 6225, the one or more instruction sets for operating the device correlated with the first digital picture are anticipated based on at least a partial match between the new digital picture and the first digital picture. Step 6225 may include any action or operation described in Step 6130 and/or Step 6135 of method 6100 as applicable.

At step 6230, the one or more instruction sets for operating the device correlated with the first digital picture are executed. Step 6230 may include any action or operation described in Step 6140 of method 6100 as applicable.

At step 6235, one or more operations defined by the one or more instruction sets for operating the device correlated with the first digital picture are performed by the device. Step 6235 may include any action or operation described in Step 6145 of method 6100 as applicable.

Referring to Fig. 33, the illustration shows an embodiment of a method 6300 for learning and/or using visual surrounding for autonomous device operation. The method can be used on a computing device or system to enable learning of a device's operation in various visual surroundings and enable autonomous device operation in similar visual surroundings. Method 6300 may include any action or operation of any of the disclosed methods such as method 6100, 6200, 6400, 6500, 6600, and/or others. Additional steps, actions, or operations can be included as

needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method 6300.

At step 6305, a first stream of digital pictures is received. Step 6305 may include any action or operation described in Step 6105 of method 6100 as applicable.

At step 6310, one or more instruction sets for operating a device are received. Step 6310 may include any action or operation described in Step 6110 of method 6100 as applicable.

At step 6315, the first stream of digital pictures is correlated with the one or more instruction sets for operating the device. Step 6315 may include any action or operation described in Step 6115 of method 6100 as applicable.

At step 6320, the first stream of digital pictures correlated with the one or more instruction sets for operating the device is stored. Step 6320 may include any action or operation described in Step 6120 of method 6100 as applicable.

At step 6325, a new stream of digital pictures is received. Step 6325 may include any action or operation described in Step 6125 of method 6100 as applicable.

At step 6330, the new stream of digital pictures is compared with the first stream of digital pictures. Step 6330 may include any action or operation described in Step 6130 of method 6100 as applicable.

At step 6335, a determination is made that there is at least a partial match between the new stream of digital pictures and the first stream of digital pictures. Step 6335 may include any action or operation described in Step 6135 of method 6100 as applicable.

At step 6340, the one or more instruction sets for operating the device correlated with the first stream of digital pictures are executed. Step 6340 may include any action or operation described in Step 6140 of method 6100 as applicable.

At step 6345, one or more operations defined by the one or more instruction sets for operating the device correlated with the first stream of digital pictures are performed by the device. Step 6345 may include any action or operation described in Step 6145 of method 6100 as applicable.

Referring to Fig. 34, the illustration shows an embodiment of a method 6400 for learning and/or using visual surrounding for autonomous device operation. The method can be used on a computing device or system to enable learning of a device's operation in various visual surroundings and enable autonomous device operation in similar visual surroundings. Method 6400 may include any action or operation of any of the disclosed methods such as method 6100, 6200, 6300, 6500, 6600, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method 6400.

At step 6405, a first digital picture is received. Step 6405 may include any action or operation described in Step 6105 of method 6100 as applicable.

At step 6410, at least one input are received, wherein the at least one input are also received by a logic circuit, and wherein the logic circuit is configured to receive inputs and produce outputs, the outputs for operating a device. Step 6410 may include any action or operation described in Step 6110 of method 6100 as applicable.

At step 6415, the first digital picture is correlated with the at least one input. Step 6415 may include any action or operation described in Step 6115 of method 6100 as applicable.

At step 6420, the first digital picture correlated with the at least one input is stored. Step 6420 may include any action or operation described in Step 6120 of method 6100 as applicable.

At step 6425, a new digital picture is received. Step 6425 may include any action or operation described in Step 6125 of method 6100 as applicable.

At step 6430, the new digital picture is compared with the first digital picture. Step 6430 may include any action or operation described in Step 6130 of method 6100 as applicable.

At step 6435, a determination is made that there is at least a partial match between the new digital picture and the first digital picture. Step 6435 may include any action or operation described in Step 6135 of method 6100 as applicable.

At step 6440, the at least one input correlated with the first digital picture are received by the logic circuit. Step 6440 may include any action or operation described in Step 6140 of method 6100 as applicable.

At step 6445, at least one operation defined by at least one output for operating the device produced by the logic circuit are performed by the device. Step 6445 may include any action or operation described in Step 6145 of method 6100 as applicable.

Referring to Fig. 35, the illustration shows an embodiment of a method 6500 for learning and/or using visual surrounding for autonomous device operation. The method can be used on a computing device or system to enable learning of a device's operation in various visual surroundings and enable autonomous device operation in similar visual surroundings. Method 6500 may include any action or operation of any of the disclosed methods such as method 6100, 6200, 6300, 6400, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method 6500.

At step 6505, a first digital picture is received. Step 6505 may include any action or operation described in Step 6105 of method 6100 as applicable.

At step 6510, at least one output are received, the at least one output transmitted from a logic circuit, wherein the logic circuit is configured to receive inputs and produce outputs, the outputs for operating a device. Step 6510 may include any action or operation described in Step 6110 of method 6100 as applicable.

At step 6515, the first digital picture is correlated with the at least one output. Step 6515 may include any action or operation described in Step 6115 of method 6100 as applicable.

At step 6520, the first digital picture correlated with the at least one output is stored. Step 6520 may include any action or operation described in Step 6120 of method 6100 as applicable.

At step 6525, a new digital picture is received. Step 6525 may include any action or operation described in Step 6125 of method 6100 as applicable.

At step 6530, the new digital picture is compared with the first digital picture. Step 6530 may include any action or operation described in Step 6130 of method 6100 as applicable.

At step 6535, a determination is made that there is at least a partial match between the new digital picture and the first digital picture. Step 6535 may include any action or operation described in Step 6135 of method 6100 as applicable.

At step 6540, at least one operation defined by the at least one output correlated with the first digital picture are performed by the device. Step 6540 may include any action or operation described in Step 6145 of method 6100 as applicable.

Referring to Fig. 36, the illustration shows an embodiment of a method 6600 for learning and/or using visual surrounding for autonomous device operation. The method can be used on a computing device or system to enable learning of a device's operation in various visual surroundings and enable autonomous device operation in similar visual surroundings. Method 6600 may include any action or operation of any of the disclosed methods such as method 6100, 6200, 6300, 6400, 6500, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method 6600.

At step 6605, a first digital picture is received. Step 6605 may include any action or operation described in Step 6105 of method 6100 as applicable.

At step 6610, at least one input are received, wherein the at least one input are also received by an actuator, and wherein the actuator is configured to receive inputs and perform motions. Step 6610 may include any action or operation described in Step 6110 of method 6100 as applicable.

At step 6615, the first digital picture is correlated with the at least one input. Step 6615 may include any action or operation described in Step 6115 of method 6100 as applicable.

At step 6620, the first digital picture correlated with the at least one input is stored. Step 6620 may include any action or operation described in Step 6120 of method 6100 as applicable.

At step 6625, a new digital picture is received. Step 6625 may include any action or operation described in Step 6125 of method 6100 as applicable.

At step 6630, the new digital picture is compared with the first digital picture. Step 6630 may include any action or operation described in Step 6130 of method 6100 as applicable.

At step 6635, a determination is made that there is at least a partial match between the new digital picture and the first digital picture. Step 6635 may include any action or operation described in Step 6135 of method 6100 as applicable.

At step 6640, the at least one input correlated with the first digital picture are received by the actuator. Step 6640 may include any action or operation described in Step 6140 of method 6100 as applicable.

At step 6645, at least one motion defined by the at least one input correlated with the first digital picture are performed by the actuator. Step 6645 may include any action or operation described in Step 6145 of method 6100 as applicable.

Referring to Fig. 37, in some exemplary embodiments, Device 98 may be or include a Computing-enabled Machine 98a. Examples of Computing-enabled Machine 98a comprise a loader, a bulldozer, an excavator, a crane, a forklift, a truck, an assembly machine, a material/object handling machine, a sorting machine, an industrial machine, a kitchen appliance, a robot, a tank, an airplane, a helicopter, a vessel, a submarine, a ground/aerial/aquatic vehicle, and/or other computing-enabled machine. In some aspects, Computing-enabled Machine 98a may itself include computing capabilities. In other aspects, computing capabilities may be included in a remote computing device (i.e. server, etc.) and provided to Computing-enabled Machine 98a (i.e. via a network, etc.). Computing-enabled Machine 98a may be operated by User 50 in person or remotely. Computing-enabled

Machine 98a may include Picture Capturing Apparatus 90 such as a motion picture, still picture, or other camera that captures one or more Digital Pictures 525 of Computing-enabled Machine's 98d surrounding. Computing-enabled Machine 98a may also include or be controlled by Logic Circuit 250 (i.e. microcontroller, etc.), Processor 11 (i.e. including any Application Program 18 running thereon, etc.), and/or other processing element that receives User's 50 (i.e. operator's, etc.) operating directions and causes desired operations with Computing-enabled Machine 98a such as moving, maneuvering, collecting, unloading, pushing, digging, lifting, and/or other operations. User 50 can interact with Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element through inputting operating directions (i.e. manipulating levers, pressing buttons, etc.) via Human-machine Interface 23 such as one or more levers or other input device. For instance, responsive to User's 50 manipulating one or more levers, Logic Circuit 250 or Processor 11 may cause Computing-enabled Machine's 98d arm with bucket to collect a load, one or more motors or other actuators to move or maneuver Computing-enabled Machine 98a, lifting system (i.e. hydraulic, pneumatic, mechanical, electrical, etc.) to lift a load, and/or arm with bucket to unload a load. Computing-enabled Machine 98a may also include or be coupled to VSADO Unit 100. VSADO Unit 100 may be embedded (i.e. integrated, etc.) into or coupled to Computing-enabled Machine's 98d Logic Circuit 250, Processor 11, and/or other processing element. VSADO Unit 100 may also be a program embedded (i.e. integrated, etc.) into or interfaced with Application Program 18 running on Processor 11 and/or other processing element. VSADO Unit 100 can obtain Instruction Sets 526 used or executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element. In some aspects, Instruction Sets 526 may include one or more inputs into or outputs from Computing-enabled Machine's 98d Logic Circuit 250 (i.e. microcontroller, etc.). In other aspects, Instruction Sets 526 may include one or more instruction sets from Computing-enabled Machine's 98d Processor's 11 registers or other components. In further aspects, Instruction Sets 526 may include one or more instruction sets used or executed in Application Program 18 running on Processor 11 and/or other processing element. VSADO Unit 100 may also optionally obtain any Extra Info 527 (i.e. time, location, computed, observed, sensory, and/or other information, etc.) related to Computing-enabled Machine's 98d operation. As User 50 operates Computing-enabled Machine 98a in various visual surroundings as shown, VSADO Unit 100 may learn Computing-enabled Machine's 98d operations in visual surroundings by correlating Digital Pictures 525 of Computing-enabled Machine's 98d surrounding with one or more Instruction Sets 526 used or executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element. Any Extra Info 527 related to Computing-enabled Machine's 98d operation may also optionally be correlated with Digital Pictures 525 of Computing-enabled Machine's 98d surrounding. VSADO Unit 100 may store this knowledge into Knowledgebase 530 (i.e. Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.). In the future, VSADO Unit 110 may compare incoming Digital Pictures 525 of Computing-enabled Machine's 98d surrounding with previously learned Digital Pictures 525 including optionally using any Extra Info 527 for enhanced decision making. If substantially similar or at least a partial match is found or determined, the Instruction Sets 526 correlated with the previously learned Digital Pictures 525 can be autonomously executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element, thereby enabling autonomous operation of Computing-enabled Machine 98a in a similar visual surrounding as in a previously learned one. For instance, Computing-enabled Machine 98a (i.e. loader, etc.) comprising VSADO Unit 100 may learn User 50-directed collecting, moving, maneuvering, lifting, and/or unloading in a visual surrounding that includes a pile of material, truck, and/or other objects with which Computing-

enabled Machine 98a may need to interact. In the future, when visual surrounding that includes same or similar objects is encountered, or when same or similar objects are detected, Computing-enabled Machine 98a may implement collecting, moving, maneuvering, lifting, and/or unloading autonomously.

Referring to Fig. 38, in some exemplary embodiments, Device 98 may be or include a Computing-enabled Machine 98a comprising or coupled to a plurality of Picture Capturing Apparatuses 90. In one example, different Picture Capturing Apparatuses 90 may capture Digital Pictures 525 of different angles of Computing-enabled Machine's 98d front. In another example, different Picture Capturing Apparatuses 90 may capture Digital Pictures 525 of the front, sides, and/or back of Computing-enabled Machine 98a. In a further example as shown, different Picture Capturing Apparatuses 90 may be placed on different sub-devices, sub-systems, or elements of Computing-enabled Machine 98a. Specifically, for instance, Picture Capturing Apparatus 90a may be placed on the roof of Computing-enabled Machine 98a (i.e. loader, etc.), Picture Capturing Apparatus 90b may be placed on the arm of Computing-enabled Machine 98a, and Picture Capturing Apparatus 90c may be placed on the bucket of Computing-enabled Machine 98a. In some designs where multiple Picture Capturing Apparatuses 90 are utilized, as User 50 operates Computing-enabled Machine 98a in various visual surroundings, VSADO Unit 100 may learn Computing-enabled Machine's 98d operations in visual surroundings by correlating collective Digital Pictures 525 of Computing-enabled Machine's 98d surrounding from multiple Picture Capturing Apparatuses 90 with one or more Instruction Sets 526 used or executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element. In other designs where multiple Picture Capturing Apparatuses 90 are utilized, multiple VSADO Units 100 may also be utilized (i.e. one VSADO Unit 100 for each Picture Capturing Apparatus 90, etc.). In such designs, as User 50 operates Computing-enabled Machine 98a in various visual surroundings, VSADO Unit 100 may learn Computing-enabled Machine's 98d operations in visual surroundings by correlating Digital Pictures 525 of Computing-enabled Machine's 98d surrounding from Picture Capturing Apparatus 90 assigned to the VSADO Unit 100 with one or more Instruction Sets 526 used or executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element. Each sub-device, sub-system, or element can, therefore, perform its own learning and/or decision making in autonomous operation.

In some embodiments, Computing-enabled Machine 98a may include a plurality of Logic Circuits 250 (i.e. microcontrollers, etc.), Processors 11, Application Programs 18, and/or other processing elements. In some aspects, each processing element may control a sub-device, sub-system, or element of Computing-enabled Machine's 98d. For example, one Processor 11 (i.e. including any Application Programs 18 running thereon, etc.) may control the moving system (i.e. drivetrain, powertrain, etc.) of Computing-enabled Machine 98a (i.e. loader), one Logic Circuit 250 may control an arm of Computing-enabled Machine 98a, and a second Logic Circuit 250 may control a bucket of Computing-enabled Machine 98a. In some designs where multiple processing elements are utilized, as User 50 operates Computing-enabled Machine 98a in various visual surroundings, VSADO Unit 100 may learn Computing-enabled Machine's 98d operations in visual surroundings by correlating Digital Pictures 525 of Computing-enabled Machine's 98d surrounding with collective one or more Instruction Sets 526 used or executed by a plurality of Logic Circuits 250, Processors 11, Application Programs 18, and/or other processing elements. In other designs where multiple processing elements are utilized, multiple VSADO Units 100 may also be utilized (i.e. one VSADO Unit 100 for each processing element, etc.). In such designs, as User 50 operates Computing-enabled Machine 98a in various visual surroundings, VSADO Unit 100 may learn Computing-enabled Machine's 98d operations in visual

surroundings by correlating Digital Pictures 525 of Computing-enabled Machine's 98d surrounding with one or more Instruction Sets 526 used or executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element assigned to the VSADO Unit 100.

In some embodiments, Computing-enabled Machine 98a (i.e. loader, etc.) may be controlled by a combination of VSADO Unit 100 and other systems and/or techniques. In some aspects, Computing-enabled Machine 98a controlled by VSADO Unit 100 may encounter a visual surrounding that has not been encountered or learned before. In such situations, User 50 and/or non-VSADO system may take control of Computing-enabled Machine's 98d operation. VSADO Unit 100 may take control again when Computing-enabled Machine 98a encounters a previously learned visual surrounding. Naturally, VSADO Unit 100 can learn Computing-enabled Machine's 98d operation in visual surroundings while User 50 and/or non-VSADO system is in control of Device 98, thereby reducing or eliminating the need for future involvement of User 50 and/or non-VSADO system. In some implementations, one User 50 can control or assist in controlling multiple Computing-enabled Machines 98d comprising VSADO Units 100. For example, User 50 can control or assist in controlling a Computing-enabled Machine 98a that may encounter a visual surrounding that has not been encountered or learned before while the Computing-enabled Machines 98d operating in previously learned visual surroundings can operate autonomously. In other aspects, Computing-enabled Machine 98a may be primarily controlled by User 50 and/or non-VSADO system. User 50 and/or non-VSADO system can release control to VSADO Unit 100 for any reason (i.e. User 50 gets tired or distracted, non-VSADO system gets stuck or cannot make a decision, etc.), at which point Computing-enabled Machine 98a can be controlled by VSADO Unit 100. In further aspects, VSADO Unit 100 may take control in certain special visual surroundings where VSADO Unit 100 may offer superior performance even if User 50 and/or non-VSADO system may generally be preferred. Once Computing-enabled Machine 98a leaves such special visual surrounding, VSADO Unit 100 may release control to User 50 and/or non-VSADO system. In general, VSADO Unit 100 can take control from, share control with, or release control to User 50, non-VSADO system, and/or other system or process at any time, under any circumstances, and remain in control for any period of time as needed.

In some embodiments, VSADO Unit 100 may control one or more sub-devices, sub-systems, or elements of Computing-enabled Machine 98a (i.e. loader) while User 50 and/or non-VSADO system may control other one or more sub-devices, sub-systems, or elements of Computing-enabled Machine 98a. For example, User 50 and/or non-VSADO system may control the moving system (i.e. drivetrain, powertrain, etc.) of Computing-enabled Machine 98a, while VSADO Unit 100 may control an arm and bucket of Computing-enabled Machine 98a. Any other combination of controlling various sub-devices, sub-systems, or elements of Computing-enabled Machine 98a by VSADO Unit 100 and User 50 and/or non-VSADO system can be implemented.

One of ordinary skill in art will understand that the features, functionalities, and embodiments described with respect to Computing-enabled Machine 98a can similarly be implemented on any computing enabled machine such as a bulldozer, an excavator, a crane, a forklift, a truck, an assembly machine, a material/object handling machine, a sorting machine, an industrial machine, a kitchen appliance, a robot, a tank, an airplane, a helicopter, a vessel, a submarine, a ground/aerial/aquatic vehicle, and/or other computing-enabled machine.

Referring to Fig. 39, in some exemplary embodiments, Device 98 may be or include a Fixture 98b. Examples of Fixture 98b comprise a fan, a light, automated blind, and/or other fixture. Fixture 98b may include Picture Capturing Apparatus 90 such as a motion picture, still picture, or other camera that captures one or more

Digital Pictures 525 of Fixture's 98b surrounding. Fixture 98b may also include or be controlled by Logic Circuit 250 (i.e. microcontroller, etc.), Processor 11 (i.e. including any Application Program 18 running thereon, etc.), and/or other processing element that receives User's 50 operating directions and causes desired operations with Fixture 98b such as setting speed of a fan, adjusting intensity of a light, adjusting angle of an automated blind, and/or other operations. User 50 can interact with Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element through inputting operating directions (i.e. pressing control buttons, switching switches, etc.) via Human-machine Interface 23 such as a controller, switch, or other input device. For instance, responsive to User's 50 pressing a control button, Logic Circuit 250 or Processor 11 may cause Fixture 98b to set a speed (i.e. in the case of a fan, etc.). Fixture 98b may also include or be coupled to VSADO Unit 100. VSADO Unit 100 may be embedded (i.e. integrated, etc.) into or coupled to Fixture's 98b Logic Circuit 250, Processor 11, and/or other processing element. VSADO Unit 100 may also be a program embedded (i.e. integrated, etc.) into or interfaced with Application Program 18 running on Processor 11 and/or other processing element. VSADO Unit 100 can obtain Instruction Sets 526 used or executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element. In some aspects, Instruction Sets 526 may include one or more inputs into or outputs from Fixture's 98b Logic Circuit 250 (i.e. microcontroller, etc.). In other aspects, Instruction Sets 526 may include one or more instruction sets from Fixture's 98b Processor's 11 registers or other components. In further aspects, Instruction Sets 526 may include one or more instruction sets used or executed in Application Program 18 running on Processor 11 and/or other processing element. VSADO Unit 100 may also optionally obtain any Extra Info 527 (i.e. time, location, computed, observed, sensory, and/or other information, etc.) related to Fixture's 98b operation. As User 50 operates Fixture 98b in a visual surrounding as shown, VSADO Unit 100 may learn Fixture's 98b operation in the visual surrounding by correlating Digital Pictures 525 of Fixture's 98b surrounding with one or more Instruction Sets 526 used or executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element. Any Extra Info 527 related to Fixture's 98b operation may also optionally be correlated with Digital Pictures 525 of Fixture's 98b surrounding. VSADO Unit 100 may store this knowledge into Knowledgebase 530 (i.e. Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.). In the future, VSADO Unit 110 may compare incoming Digital Pictures 525 of Fixture's 98b surrounding with previously learned Digital Pictures 525 including optionally using any Extra Info 527 for enhanced decision making. If substantially similar or at least a partial match is found or determined, the Instruction Sets 526 correlated with the previously learned Digital Pictures 525 can be autonomously executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element, thereby enabling autonomous operation of Fixture 98b in a similar visual surrounding as in a previously learned one. For instance, Fixture 98b (i.e. ceiling fan, etc.) comprising VSADO Unit 100 may learn User's 50 setting speed of Fixture 98b in a visual surrounding that includes User 50 entering or being present in a room. In the future, when visual surrounding that includes User 50 entering or being present in the room, or when User 50 or his/her body part (i.e. face, etc.) is detected, Fixture 98b may implement setting of its speed autonomously. In some aspects, Fixture 98b comprising VSADO Unit 100 may engage autonomous operation (i.e. autonomous fan speed setting, etc.) if a specific person is detected by using facial recognition, thereby personalizing the operation of Fixture 98b. In other aspects, Fixture 98b may engage autonomous operation (i.e. autonomous fan speed setting, etc.) if any person is detected by using person or object recognition.

Referring to Fig. 40, in some exemplary embodiments, Device 98 may be or include a Control Device 98c. Examples of Control Device 98c comprise a thermostat, a control panel, a remote or other controller, and/or other control device. Control Device 98c may include Picture Capturing Apparatus 90 such as a motion picture, still picture, or other camera that captures one or more Digital Pictures 525 of Control Device's 98c surrounding. Control Device 98c may also include Logic Circuit 250 (i.e. microcontroller, etc.), Processor 11 (i.e. including any Application Program 18 running thereon, etc.), and/or other processing element that receives User's 50 operating directions and causes desired operations on a device or system controlled by Control Device 98c such as regulating temperature of an air conditioning system, and/or other operations. User 50 can interact with Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element through inputting operating directions (i.e. pressing control buttons, etc.) via Human-machine Interface 23 such as a control panel or other input device. For instance, responsive to User's 50 pressing a control button, Logic Circuit 250 or Processor 11 may cause Control Device 98c to increase or decrease a temperature of an air conditioning system. Control Device 98c may also include or be coupled to VSADO Unit 100. VSADO Unit 100 may be embedded (i.e. integrated, etc.) into or coupled to Control Device's 98c Logic Circuit 250, Processor 11, and/or other processing element. VSADO Unit 100 may also be a program embedded (i.e. integrated, etc.) into or interfaced with Application Program 18 running on Processor 11 and/or other processing element. VSADO Unit 100 can obtain Instruction Sets 526 used or executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element. In some aspects, Instruction Sets 526 may include one or more inputs into or outputs from Control Device's 98c Logic Circuit 250 (i.e. microcontroller, etc.). In other aspects, Instruction Sets 526 may include one or more instruction sets from Control Device's 98c Processor's 11 registers or other components. In further aspects, Instruction Sets 526 may include one or more instruction sets used or executed in Application Program 18 running on Processor 11 and/or other processing element. VSADO Unit 100 may also optionally obtain any Extra Info 527 (i.e. time, location, computed, observed, sensory, and/or other information, etc.) related to Control Device's 98c operation. As User 50 operates Control Device 98c in a visual surrounding as shown, VSADO Unit 100 may learn Control Device's 98c operation in the visual surrounding by correlating Digital Pictures 525 of Control Device's 98c surrounding with one or more Instruction Sets 526 used or executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element. Any Extra Info 527 related to Control Device's 98c operation may also optionally be correlated with Digital Pictures 525 of Control Device's 98c surrounding. VSADO Unit 100 may store this knowledge into Knowledgebase 530 (i.e. Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.). In the future, VSADO Unit 110 may compare incoming Digital Pictures 525 of Control Device's 98c surrounding with previously learned Digital Pictures 525 including optionally using any Extra Info 527 for enhanced decision making. If substantially similar or at least a partial match is found or determined, the Instruction Sets 526 correlated with the previously learned Digital Pictures 525 can be autonomously executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element, thereby enabling autonomous operation of Control Device 98c in a similar visual surrounding as in a previously learned one. For instance, Control Device 98c comprising VSADO Unit 100 may learn User's 50 setting temperature of an air conditioning system controlled by Control Device 98c in a visual surrounding that includes User 50 entering or being present in a room. In the future, when visual surrounding that includes User 50 entering or being present in the room, or when User 50 or his/her body part (i.e. face, etc.) is detected, Control Device 98c may

implement setting temperature of the air conditioning system autonomously. In some aspects, Control Device 98c may engage autonomous operation (i.e. autonomous temperature setting of an air conditioning system, etc.) if a specific person is detected by using facial recognition, thereby personalizing the operation of Control Device 98c. In other aspects, Control Device 98c may engage autonomous operation (i.e. autonomous temperature setting of an air conditioning system, etc.) if any person is detected by using person or object recognition.

Referring to Fig. 41, in some exemplary embodiments, Device 98 may be or include a Smartphone 98d. Examples of Smartphone 98d comprise Apple iPhone, Samsung Galaxy, Microsoft Lumia, and/or other smartphone. Smartphone 98d may include Picture Capturing Apparatus 90 such as a motion picture, still picture, or other camera that captures one or more Digital Pictures 525 of Smartphone's 98a surrounding. Smartphone 98d may include Processor 11 and one or more Application Programs 18 such as a phone control application that receives User's 50 operating directions and causes desired operations with Smartphone 98d such as making a call, ending a call, increasing volume, setting Smartphone 98d on vibrate mode, and/or other operations. User 50 can interact with Processor 11 and/or Application Program 18 through inputting operating directions (i.e. touching touchscreen elements, etc.) via Human-machine Interface 23 such as a touchscreen or other input device. For instance, responsive to User's 50 touching a touchscreen element, Processor 11 and/or Application Program 18 may cause Smartphone 98d to go into a vibrate mode. Smartphone 98d may also include or be coupled to VSADO Unit 100. VSADO Unit 100 may be embedded (i.e. integrated, etc.) into or coupled to Smartphone's 98a Processor 11 and/or other processing element. VSADO Unit 100 may also be a program embedded (i.e. integrated, etc.) into or interfaced with Application Program 18 running on Processor 11 and/or other processing element. VSADO Unit 100 can obtain Instruction Sets 526 used or executed by Processor 11, Application Program 18, and/or other processing element. In some aspects, Instruction Sets 526 may include one or more instruction sets used or executed in Application Program 18 running on Processor 11 and/or other processing element. In other aspects, Instruction Sets 526 may include one or more instruction sets from Smartphone's 98a Processor's 11 registers or other components. VSADO Unit 100 may also optionally obtain any Extra Info 527 (i.e. time, location, computed, observed, sensory, and/or other information, etc.) related to Smartphone's 98a operation. As User 50 operates Smartphone 98d in a visual surroundings as shown, VSADO Unit 100 may learn Smartphone's 98a operation in the visual surrounding by correlating Digital Pictures 525 of Smartphone's 98a surrounding with one or more Instruction Sets 526 used or executed by Processor 11, Application Program 18, and/or other processing element. Any Extra Info 527 related to Smartphone's 98a operation may also optionally be correlated with Digital Pictures 525 of Smartphone's 98a surrounding. VSADO Unit 100 may store this knowledge into Knowledgebase 530 (i.e. Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.). In the future, VSADO Unit 110 may compare incoming Digital Pictures 525 of Smartphone's 98a surrounding with previously learned Digital Pictures 525 including optionally using any Extra Info 527 for enhanced decision making. If substantially similar or at least a partial match is found or determined, the Instruction Sets 526 correlated with the previously learned Digital Pictures 525 can be autonomously executed by Processor 11, Application Program 18, and/or other processing element, thereby enabling autonomous operation of Smartphone 98d in a similar visual surrounding as in a previously learned one. For instance, Smartphone 98d comprising VSADO Unit 100 may learn User's 50 setting of Smartphone 98d on vibrate mode in a visual surrounding that includes a classroom. In the future, when visual surrounding that includes a classroom is encountered, or when classroom is detected,

Smartphone 98d may implement vibrate setting autonomously. In some aspects, similar functionality can be utilized in visual surroundings that include a house of worship, cemetery, and/or others.

In some embodiments, VSADO Unit 100 can be used to enable Smartphone 98d, computer, and/or application to learn User's 50 movements for interacting with or controlling Smartphone 98d, computer, and/or application. In one example, while viewing a web page in a web browser running on Smartphone 98d, User 50 may perform a head nod during or after which User 50 may scroll down the web page. Smartphone 98d comprising VSADO Unit 100 may learn User's 50 scrolling of a web page in a visual surrounding that includes User 50 performing a head nod. In the future, when visual surrounding that includes User 50 performing a head nod is encountered or detected, Smartphone 98d may implement scrolling of a web page in a web browser autonomously. In another example, while operating a user controllable object (i.e. avatar, etc.) in a computer game running on Smartphone 98d, User 50 may lean right during or after which User 50 may direct the user controllable object to turn or steer right. Smartphone 98d comprising VSADO Unit 100 may learn User's 50 directing the user controllable object to turn or steer right in a visual surrounding that includes User 50 leaning right. In the future, when visual surrounding that includes User 50 leaning right is encountered or detected, Smartphone 98d may implement directing the user controllable object to turn or steer right in a computer game autonomously. Therefore, VSADO Unit 100 can spontaneously learn both User's 50 movements and Instruction Sets 526 implementing an operation without User 50 needing to program, manually designate, or otherwise assign the movements to Instruction Sets 526 implementing the operation. Such functionality enables learning of User 50-chosen movements and User 50-chosen operations seamlessly as User 50 operates a device, application, and/or object thereof in real life situations without the need for special training sessions. Any User's 50 movements can be utilized examples of which include moving head, moving facial parts (i.e. eyes, lips, etc.), moving shoulders, moving hands, moving hand parts (i.e. fingers, etc.), moving body, moving body parts (i.e. arms, legs, etc.), and/or others. Any of the functionalities described with respect to Smartphone 98d similarly apply to any computer or computing enabled device.

It must be noted that as used herein and in the appended claims, the singular forms "a", "an", and "the" include plural referents unless the context clearly dictates otherwise.

A number of embodiments have been described herein. While this disclosure contains many specific implementation details, these should not be construed as limitations on the scope of any inventions or of what may be claimed, but rather as descriptions of features specific to particular embodiments. It should be understood that various modifications can be made without departing from the spirit and scope of the invention. The logic flows depicted in the figures do not require the particular order shown, or sequential order, to achieve desirable results. In addition, other or additional steps, elements, or connections can be included, or some of the steps, elements, or connections can be eliminated, or a combination thereof can be utilized in the described flows, illustrations, or descriptions. Further, the various aspects of the disclosed devices, apparatuses, systems, and/or methods can be combined in whole or in part with each other to produce additional implementations. Moreover, separation of various components in the embodiments described herein should not be understood as requiring such separation in all embodiments, and it should be understood that the described components can generally be integrated together in a single software product or packaged into multiple software products. Accordingly, other embodiments are within the scope of the following claims.

PTO/AIA/14 (11-15)
Approved for use through 04/30/2017. OMB 0651-0032
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	
	Application Number	

Title of Invention	MACHINE LEARNING FOR COMPUTING ENABLED SYSTEMS AND/OR DEVICES

The application data sheet is part of the provisional or nonprovisional application for which it is being submitted. The following form contains the bibliographic data arranged in a format specified by the United States Patent and Trademark Office as outlined in 37 CFR 1.76.
This document may be completed electronically and submitted to the Office in electronic format using the Electronic Filing System (EFS) or the document may be printed and included in a paper filed application.

Secrecy Order 37 CFR 5.2:

☐ Portions or all of the application associated with this Application Data Sheet may fall under a Secrecy Order pursuant to 37 CFR 5.2 (Paper filers only. Applications that fall under Secrecy Order may not be filed electronically.)

Inventor Information:

Inventor 1 Remove

Legal Name

Prefix	Given Name	Middle Name	Family Name	Suffix
	Jasmin		Cosic	

Residence Information (Select One) ● US Residency Non US Residency Active US Military Service

City	Miami	State/Province	FL	Country of Residence	US

Mailing Address of Inventor:

Address 1	108 Woodbury Street		
Address 2			
City	Pawtucket	State/Province	RI
Postal Code	02861	Country	US

All Inventors Must Be Listed - Additional Inventor Information blocks may be generated within this form by selecting the **Add** button. Add

Correspondence Information:

Enter either Customer Number or complete the Correspondence Information section below. For further information see 37 CFR 1.33(a).

☐ **An Address is being provided for the correspondence Information of this application.**

Customer Number	116094		
Email Address	cpapc29@hotmail.com	Add Email	Remove Email

Application Information:

Title of the Invention	MACHINE LEARNING FOR COMPUTING ENABLED SYSTEMS AND/OR DEVICES		
Attorney Docket Number		Small Entity Status Claimed	☒
Application Type	Nonprovisional		
Subject Matter	Utility		
Total Number of Drawing Sheets (if any)	41	Suggested Figure for Publication (if any)	2

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	
	Application Number	

Title of Invention	MACHINE LEARNING FOR COMPUTING ENABLED SYSTEMS AND/OR DEVICES

Filing By Reference:

Only complete this section when filing an application by reference under 35 U.S.C. 111(c) and 37 CFR 1.57(a). Do not complete this section if application papers including a specification and any drawings are being filed. Any domestic benefit or foreign priority information must be provided in the appropriate section(s) below (i.e., "Domestic Benefit/National Stage Information" and "Foreign Priority Information").

For the purposes of a filing date under 37 CFR 1.53(b), the description and any drawings of the present application are replaced by this reference to the previously filed application, subject to conditions and requirements of 37 CFR 1.57(a).

Application number of the previously filed application	Filing date (YYYY-MM-DD)	Intellectual Property Authority or Country

Publication Information:

☐ Request Early Publication (Fee required at time of Request 37 CFR 1.219)

☒ **Request Not to Publish.** I hereby request that the attached application not be published under 35 U.S.C. 122(b) and certify that the invention disclosed in the attached application **has not and will not** be the subject of an application filed in another country, or under a multilateral international agreement, that requires publication at eighteen months after filing.

Representative Information:

Representative information should be provided for all practitioners having a power of attorney in the application. Providing this information in the Application Data Sheet does not constitute a power of attorney in the application (see 37 CFR 1.32).
Either enter Customer Number or complete the Representative Name section below. If both sections are completed the customer Number will be used for the Representative Information during processing.

Please Select One:	● Customer Number	US Patent Practitioner	○ Limited Recognition (37 CFR 11.9)
Customer Number			

Domestic Benefit/National Stage Information:

This section allows for the applicant to either claim benefit under 35 U.S.C. 119(e), 120, 121, 365(c), or 386(c) or indicate National Stage entry from a PCT application. Providing benefit claim information in the Application Data Sheet constitutes the specific reference required by 35 U.S.C. 119(e) or 120, and 37 CFR 1.78.
When referring to the current application, please leave the "Application Number" field blank.

Prior Application Status	Pending		Remove
Application Number	Continuity Type	Prior Application Number	Filing or 371(c) Date (YYYY-MM-DD)
	Continuation of	15822150	2017-11-26

Additional Domestic Benefit/National Stage Data may be generated within this form by selecting the **Add** button. Add

PTO/AIA/14 (11-15)
Approved for use through 04/30/2017. OMB 0651-0032
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	
	Application Number	
Title of Invention	MACHINE LEARNING FOR COMPUTING ENABLED SYSTEMS AND/OR DEVICES	

Foreign Priority Information:

This section allows for the applicant to claim priority to a foreign application. Providing this information in the application data sheet constitutes the claim for priority as required by 35 U.S.C. 119(b) and 37 CFR 1.55. When priority is claimed to a foreign application that is eligible for retrieval under the priority document exchange program (PDX)[i] the information will be used by the Office to automatically attempt retrieval pursuant to 37 CFR 1.55(i)(1) and (2). Under the PDX program, applicant bears the ultimate responsibility for ensuring that a copy of the foreign application is received by the Office from the participating foreign intellectual property office, or a certified copy of the foreign priority application is filed, within the time period specified in 37 CFR 1.55(g)(1).

Application Number	Country[i]	Filing Date (YYYY-MM-DD)	Remove Access Code[i] (if applicable)

Additional Foreign Priority Data may be generated within this form by selecting the **Add** button. Add

Statement under 37 CFR 1.55 or 1.78 for AIA (First Inventor to File) Transition Applications

☐ This application (1) claims priority to or the benefit of an application filed before March 16, 2013 and (2) also contains, or contained at any time, a claim to a claimed invention that has an effective filing date on or after March 16, 2013.
NOTE: By providing this statement under 37 CFR 1.55 or 1.78, this application, with a filing date on or after March 16, 2013, will be examined under the first inventor to file provisions of the AIA.

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	
	Application Number	
Title of Invention	MACHINE LEARNING FOR COMPUTING ENABLED SYSTEMS AND/OR DEVICES	

Authorization or Opt-Out of Authorization to Permit Access:

When this Application Data Sheet is properly signed and filed with the application, applicant has provided written authority to permit a participating foreign intellectual property (IP) office access to the instant application-as-filed (see paragraph A in subsection 1 below) and the European Patent Office (EPO) access to any search results from the instant application (see paragraph B in subsection 1 below).

Should applicant choose not to provide an authorization identified in subsection 1 below, applicant **must opt-out** of the authorization by checking the corresponding box A or B or both in subsection 2 below.

NOTE: This section of the Application Data Sheet is **ONLY** reviewed and processed with the **INITIAL** filing of an application. After the initial filing of an application, an Application Data Sheet cannot be used to provide or rescind authorization for access by a foreign IP office(s). Instead, Form PTO/SB/39 or PTO/SB/69 must be used as appropriate.

1. Authorization to Permit Access by a Foreign Intellectual Property Office(s)

A. Priority Document Exchange (PDX) - Unless box A in subsection 2 (opt-out of authorization) is checked, the undersigned hereby **grants the USPTO authority** to provide the European Patent Office (EPO), the Japan Patent Office (JPO), the Korean Intellectual Property Office (KIPO), the State Intellectual Property Office of the People's Republic of China (SIPO), the World Intellectual Property Organization (WIPO), and any other foreign intellectual property office participating with the USPTO in a bilateral or multilateral priority document exchange agreement in which a foreign application claiming priority to the instant patent application is filed, access to: (1) the instant patent application-as-filed and its related bibliographic data, (2) any foreign or domestic application to which priority or benefit is claimed by the instant application and its related bibliographic data, and (3) the date of filing of this Authorization. See 37 CFR 1.14(h)(1).

B. Search Results from U.S. Application to EPO - Unless box B in subsection 2 (opt-out of authorization) is checked, the undersigned hereby **grants the USPTO authority** to provide the EPO access to the bibliographic data and search results from the instant patent application when a European patent application claiming priority to the instant patent application is filed. See 37 CFR 1.14(h)(2).

The applicant is reminded that the EPO's Rule 141(1) EPC (European Patent Convention) requires applicants to submit a copy of search results from the instant application without delay in a European patent application that claims priority to the instant application.

2. Opt-Out of Authorizations to Permit Access by a Foreign Intellectual Property Office(s)

☒ A. Applicant **DOES NOT** authorize the USPTO to permit a participating foreign IP office access to the instant application-as-filed. If this box is checked, the USPTO will not be providing a participating foreign IP office with any documents and information identified in subsection 1A above.

☒ B. Applicant **DOES NOT** authorize the USPTO to transmit to the EPO any search results from the instant patent application. If this box is checked, the USPTO will not be providing the EPO with search results from the instant application.

NOTE: Once the application has published or is otherwise publicly available, the USPTO may provide access to the application in accordance with 37 CFR 1.14.

PTO/AIA/14 (11-15)
Approved for use through 04/30/2017. OMB 0651-0032
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	
	Application Number	
Title of Invention	MACHINE LEARNING FOR COMPUTING ENABLED SYSTEMS AND/OR DEVICES	

Applicant Information:

Providing assignment information in this section does not substitute for compliance with any requirement of part 3 of Title 37 of CFR to have an assignment recorded by the Office.

Applicant 1 Remove

If the applicant is the inventor (or the remaining joint inventor or inventors under 37 CFR 1.45), this section should not be completed. The information to be provided in this section is the name and address of the legal representative who is the applicant under 37 CFR 1.43; or the name and address of the assignee, person to whom the inventor is under an obligation to assign the invention, or person who otherwise shows sufficient proprietary interest in the matter who is the applicant under 37 CFR 1.46. If the applicant is an applicant under 37 CFR 1.46 (assignee, person to whom the inventor is obligated to assign, or person who otherwise shows sufficient proprietary interest) together with one or more joint inventors, then the joint inventor or inventors who are also the applicant should be identified in this section.

Clear

Assignee	Legal Representative under 35 U.S.C. 117	Joint Inventor
Person to whom the inventor is obligated to assign.	Person who shows sufficient proprietary interest	

If applicant is the legal representative, indicate the authority to file the patent application, the inventor is:

Name of the Deceased or Legally Incapacitated Inventor:

If the Applicant is an Organization check here. ☐

Prefix	**Given Name**	Middle Name	**Family Name**	Suffix

Mailing Address Information For Applicant:

Address 1			
Address 2			
City		**State/Province**	
Country		Postal Code	
Phone Number		Fax Number	
Email Address			

Additional Applicant Data may be generated within this form by selecting the Add button. Add

Assignee Information including Non-Applicant Assignee Information:

Providing assignment information in this section does not substitute for compliance with any requirement of part 3 of Title 37 of CFR to have an assignment recorded by the Office.

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	
	Application Number	
Title of Invention	MACHINE LEARNING FOR COMPUTING ENABLED SYSTEMS AND/OR DEVICES	

Assignee 1

Complete this section if assignee information, including non-applicant assignee information, is desired to be included on the patent application publication. An assignee-applicant identified in the "Applicant Information" section will appear on the patent application publication as an applicant. For an assignee-applicant, complete this section only if identification as an assignee is also desired on the patent application publication.

Remove

If the Assignee or Non-Applicant Assignee is an Organization check here. ☐

Prefix	**Given Name**	Middle Name	**Family Name**	Suffix

Mailing Address Information For Assignee including Non-Applicant Assignee:

Address 1			
Address 2			
City		**State/Province**	
Country i		Postal Code	
Phone Number		Fax Number	
Email Address			

Additional Assignee or Non-Applicant Assignee Data may be generated within this form by selecting the Add button. Add

Signature:

Remove

NOTE: This Application Data Sheet must be signed in accordance with 37 CFR 1.33(b). **However, if this Application Data Sheet is submitted with the INITIAL filing of the application and either box A or B is not checked in subsection 2 of the "Authorization or Opt-Out of Authorization to Permit Access" section, then this form must also be signed in accordance with 37 CFR 1.14(c).**

This Application Data Sheet **must** be signed by a patent practitioner if one or more of the applicants is a **juristic entity** (e.g., corporation or association). If the applicant is two or more joint inventors, this form must be signed by a patent practitioner, **all** joint inventors who are the applicant, or one or more joint inventor-applicants who have been given power of attorney (e.g., see USPTO Form PTO/AIA/81) on behalf of **all** joint inventor-applicants.

See 37 CFR 1.4(d) for the manner of making signatures and certifications.

Signature	/Jasmin Cosic/			Date (YYYY-MM-DD)	2019-09-26
First Name	Jasmin	Last Name	Cosic	Registration Number	

Additional Signature may be generated within this form by selecting the Add button. Add

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	
	Application Number	

Title of Invention	MACHINE LEARNING FOR COMPUTING ENABLED SYSTEMS AND/OR DEVICES

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1 The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these records.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3 A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent CooperationTreaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

CLAIMS

1. A system comprising:

one or more processors; and

one or more non-transitory machine readable media storing machine readable code that, when executed by the one or more processors, causes the one or more processors to perform at least:

accessing a first correlation including a first one or more digital pictures correlated with a first one or more instruction sets for operating a first device.

2. The system of claim 1, wherein the first correlation is learned in a learning process that includes:

receiving the first one or more digital pictures; and

receiving the first one or more instruction sets for operating the first device.

3. The system of claim 2, wherein the learning process further includes:

correlating the first one or more digital pictures with the first one or more instruction sets for operating the first device; and

storing the first correlation.

4. The system of claim 1, wherein:

an element of the first correlation is deleted, modified, or manipulated after the first correlation is generated, or

an element is inserted into the first correlation after the first correlation is generated.

5. The system of claim 1, wherein the first one or more digital pictures include: one or more still digital pictures, or one or more motion digital pictures.

6. The system of claim 1, wherein at least a portion of the first correlation is learned in a learning process while a user operates the first device.

7. The system of claim 1, wherein the machine readable code, when executed by the one or more processors, causes the one or more processors to further perform at least:

receiving a second one or more digital pictures;

determining the first one or more instruction sets for operating the first device based on at least partial match between the second one or more digital pictures and the first one or more digital pictures; and

at least in response to the determining, causing the first device or a second device to perform one or more operations defined by the first one or more instruction sets for operating the first device.

8. The system of claim 7, wherein the determining the first one or more instruction sets for operating the first device based on the at least partial match between the second one or more digital pictures and the first one or more digital pictures includes:

determining that a number of at least partially matching portions of the second one or more digital pictures and portions of the first one or more digital pictures exceeds a threshold number, or

determining that a percentage of at least partially matching portions of the second one or more digital pictures and portions of the first one or more digital pictures exceeds a threshold percentage.

9. The system of claim 8, wherein the portions of the second one or more digital pictures include portions of the second one or more digital pictures that depict recognized objects, and wherein the portions of the first one or more digital pictures include portions of the first one or more digital pictures that depict recognized objects.

10. The system of claim 1, wherein the first one or more digital pictures are received from a picture capturing apparatus included in the first device, and wherein the machine readable code, when executed by the one or more processors, causes the one or more processors to further perform at least:

receiving a second one or more digital pictures from a picture capturing apparatus included in a second device;

determining the first one or more instruction sets for operating the first device based on at least partial match between the second one or more digital pictures and the first one or more digital pictures; and

at least in response to the determining, causing the second device to perform one or more operations defined by: the first one or more instruction sets for operating the first device, or modified first one or more instruction sets for operating the first device.

11. The system of claim 1, wherein the first correlation is included in a first knowledge cell of a knowledgebase, and wherein the knowledgebase further includes a second knowledge cell including a second correlation, and wherein the second correlation includes a second one or more digital pictures correlated with a second one or more instruction sets for operating: the first device, or a second device.

12. The system of claim 1, wherein the first correlation is included in a knowledgebase, and wherein the knowledgebase further includes a second correlation including a second one or more digital pictures correlated with a second one or more instruction sets for operating: the first device, or a second device, and wherein the first correlation is connected with the second correlation by a connection.

13. The system of claim 1, wherein the first correlation is included in a knowledgebase, and wherein the knowledgebase further includes a second correlation including a second one or more digital pictures correlated with a second one or more instruction sets for operating the first device, and wherein at least a portion of the first correlation is learned in a first learning process that includes operating the first device at least partially by a first user, and wherein at least a portion of the second correlation is learned in a second learning process that includes operating the first device at least partially by the first user.

14. The system of claim 1, wherein the first correlation is included in a knowledgebase,, and wherein the knowledgebase further includes a second correlation including a second one or more digital pictures correlated with a second one or more instruction sets for operating the first device, and wherein at least a portion of the first correlation is learned in a first learning process that includes operating the first device at least partially by a first user, and wherein at least a portion of the second correlation is learned in a second learning process that includes operating the first device at least partially by a second user.

15. The system of claim 1, wherein the first correlation is included in a knowledgebase, and wherein the knowledgebase further includes a second correlation including a second one or more digital pictures correlated with a second one or more instruction sets for operating a second device, and wherein at least a portion of the first correlation is learned in a first learning process that

includes operating the first device at least partially by a first user, and wherein at least a portion of the second correlation is learned in a second learning process that includes operating the second device at least partially by a second user.

16. The system of claim 1, wherein, to correlate the first one or more digital pictures with the first one or more instruction sets for operating the first device, a determination is made that the first one or more instruction sets for operating the first device temporally correspond to the first one or more digital pictures.

17. The system of claim 1, wherein at least some elements of the system are included in: the first device, another device, or multiple devices, and wherein the one or more processors include: one or more microcontrollers, one or more computing devices, or one or more electronic devices, and wherein the first device includes: a robot, a vehicle, an appliance, an electronic device, or a mechanical machine, and wherein the first correlation is stored in: at least one non-transitory machine readable medium of the one or more non-transitory machine readable media, or another one or more non-transitory machine readable media, and wherein an instruction set of the first one or more instruction sets for operating the first device includes at least one selected from the group comprising: only one instruction, a plurality of instructions, one or more inputs, one or more commands, one or more computer commands, one or more keywords, one or more symbols, one or more operators, one or more variables, one or more values, one or more objects or object references, one or more data

structures or data structure references, one or more functions or function references, one or more parameters, one or more signals, one or more characters, one or more digits, one or more user operating directions, one or more user directions, one or more user inputs, one or more representations of one or more user actions, one or more representations of one or more user clicks, one or more binary bits, one or more assembly language commands, one or more states, one or more codes, one or more data, and one or more information.

18. The system of claim 1, wherein the first one or more instruction sets for operating the first device include one or more information about one or more states of the first device.

19. A system comprising:

means for processing; and

means for storing machine readable code that, when executed by the means for processing, causes the means for processing to perform at least:

accessing a first correlation including a first one or more digital pictures correlated with a first one or more instruction sets for operating a first device.

20. A method implemented using a computing system that includes one or more processors, the method comprising:

accessing a first correlation including a first one or more digital pictures correlated with a first one or more instruction sets for operating a first device.

Electronic Patent Application Fee Transmittal

Application Number:	
Filing Date:	
Title of Invention:	MACHINE LEARNING FOR COMPUTING ENABLED SYSTEMS AND/OR DEVICES
First Named Inventor/Applicant Name:	Jasmin Cosic
Filer:	Jasmin Cosic
Attorney Docket Number:	

Filed as Small Entity

Filing Fees for **Utility under 35 USC 111(a)**

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Basic Filing:				
UTILITY FILING FEE (ELECTRONIC FILING)	4011	1	75	75
UTILITY SEARCH FEE	2111	1	330	330
UTILITY EXAMINATION FEE	2311	1	380	380
Pages:				
UTILITY APPL SIZE FEE PER 50 SHEETS >100	2081	1	200	200
Claims:				
Miscellaneous-Filing:				
Petition:				

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Patent-Appeals-and-Interference:				
Post-Allowance-and-Post-Issuance:				
Extension-of-Time:				
Miscellaneous:				
	Total in USD ($)			**985**

Electronic Acknowledgement Receipt

EFS ID:	37293386
Application Number:	16584736
International Application Number:	
Confirmation Number:	3341
Title of Invention:	MACHINE LEARNING FOR COMPUTING ENABLED SYSTEMS AND/OR DEVICES
First Named Inventor/Applicant Name:	Jasmin Cosic
Customer Number:	116094
Filer:	Jasmin Cosic
Filer Authorized By:	
Attorney Docket Number:	
Receipt Date:	26-SEP-2019
Filing Date:	
Time Stamp:	19:40:34
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	yes
Payment Type	CARD
Payment was successfully received in RAM	$985
RAM confirmation Number	E20199PJ43152861
Deposit Account	
Authorized User	

The Director of the USPTO is hereby authorized to charge indicated fees and credit any overpayment as follows:

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1	Abstract	ABSTRACT.pdf	6584 afc26a2b0dd81966a5402b8951bde28105f83a1e	no	1
Warnings:					
Information:					
2	Oath or Declaration filed	DECLARATION.pdf	399165 2c26cf8c3bedd5f4d6d51bb410de245233f59eaa	no	1
Warnings:					
Information:					
3	Drawings-only black and white line drawings	DRAWINGS.pdf	1676888 182f01cd770a052cd58f2e3555488d226d50b524	no	41
Warnings:					
The page size in the PDF is too large. The pages should be 8.5 x 11 or A4. If this PDF is submitted, the pages will be resized upon entry into the Image File Wrapper and may affect subsequent processing					
Information:					
4	Specification	SPECIFICATION.pdf	911058 f342742f11f821b6c3b59fef67549c160ba627db	no	144
Warnings:					
Information:					
5	Application Data Sheet	ADS.pdf	1826245 42f1036ffb92346749b8ceaa6d474bcf4a637ef7	no	8
Warnings:					
Information:					
6	Claims	CLAIMS.pdf	27758 4ae4efdf9f3c3656d1d19d34e158f8bdcd5d44c4	no	8
Warnings:					

Information:					
7	Fee Worksheet (SB06)	fee-info.pdf	36432 daa5035bbb41e4e5e6a4a2ea4051bf1b6def3826	no	2
Warnings:					
Information:					
		Total Files Size (in bytes):	4884130		

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.
National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.
New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

PTO/SB/08a (01-10)
Approved for use through 07/31/2012. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it contains a valid OMB control number.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS Remove

Examiner Initial*	Cite No	Patent Number	Kind Code1	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	4370707		1983-01-25	Phillips , et al.	
	2	4730315		1988-03-08	Saito , et al.	
	3	4860203		1989-08-22	Corrigan , et al.	
	4	5602982		1997-02-11	Judd , et al.	
	5	6026234		2000-02-15	Hanson , et al.	
	6	6088731		2000-07-11	Kiraly , et al.	
	7	6106299		2000-08-22	Ackermann , et al.	
	8	6126330		2000-10-03	Knight	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	9	6314558		2001-11-06	Angel , et al.	
	10	6643842		2003-11-04	Angel , et al.	
	11	6728689		2004-04-27	Drissi , et al.	
	12	6735632		2004-05-11	Kiraly , et al.	
	13	6801912		2004-10-05	Moskowitz , et al.	
	14	6915105		2005-07-05	Masuda	
	15	6973446		2005-12-06	Mamitsuka , et al.	
	16	7017153		2006-03-21	Gouriou , et al.	
	17	7052277		2006-05-30	Kellman	
	18	7222127		2007-05-22	Bem , et al.	
	19	7240335		2007-07-03	Angel , et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	20	7249349		2007-07-24	Hundt , et al.	
	21	7424705		2008-09-09	Lewis , et al.	
	22	7478371		2009-01-13	Gove	
	23	7484205		2009-01-27	Venkatapathy	
	24	7721218		2010-05-18	Awe , et al.	
	25	7765537		2010-07-27	Havin , et al.	
	26	7797259		2010-09-14	Jiang , et al.	
	27	7840060		2010-11-23	Podilchuk	
	28	7925984		2011-04-12	Awe , et al.	
	29	7987144		2011-07-26	Drissi , et al.	
	30	8005828		2011-08-23	Buchner , et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	31	8019699		2011-09-13	Baxter	
	32	8078556		2011-12-13	Adi , et al.	
	33	8090669		2012-01-03	Shahani , et al.	
	34	8137112		2012-03-20	Woolf , et al.	
	35	8166463		2012-04-24	Gill , et al.	
	36	8195674		2012-06-05	Bem , et al.	
	37	8196119		2012-06-05	Gill , et al.	
	38	8244730		2012-08-14	Gupta	
	39	8251704		2012-08-28	Woolf , et al.	
	40	8266608		2012-09-11	Hecht , et al.	
	41	8364612		2013-01-29	Van Gael , et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	42	8386401		2013-02-26	Virkar , et al.	
	43	8397227		2013-03-12	Fan , et al.	
	44	8464225		2013-06-11	Greifeneder	
	45	8549359		2013-10-01	Zheng	
	46	8589414		2013-11-19	Waite , et al.	
	47	8655260		2014-02-18	Yueh , et al.	
	48	8667472		2014-03-04	Molinari	

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS Remove

Examiner Initial*	Cite No	Publication Number	Kind Code[1]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	20120167057		2012-06-28	Schmich; Christopher P. ; et al.	
	2	20120150773		2012-06-14	DiCorpo; Phillip ; et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	3	20080254429		2008-10-16	Woolf; Susan D. ; et al.	
	4	20040117771		2004-06-17	Venkatapathy, Ramanathan	
	5	20130238533		2013-09-12	VIRKAR; Hemant ; et al.	
	6	20060190930		2006-08-24	Hecht; Daniel M. ; et al.	
	7	20070006159		2007-01-04	Hecht; Daniel Michael ; et al.	
	8	20080281764		2008-11-13	Baxter; Jonathan	
	9	20100138370		2010-06-03	Wu; Wang ; et al.	
	10	20110218672		2011-09-08	Maisonnier; Bruno ; et al.	

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number[3]	Country Code[2] i	Kind Code[4]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T[5]
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T5
	1	About Event Tracing, retrieved from <URL: http://msdn.microsoft.com/en-us/library/aa363668(d=default,l=en-us, v=vs.85).aspx> on Jan 12, 2014, 2 pages	
	2	Add dynamic Java code to your application, retrieved from <URL: http://www.javaworld.com/article/2071777/design-patterns/add-dynamic-java-code-to-your-application.html> on May 13, 2014, 6 pages	
	3	BCEL, retrieved from <URL: http://commons.apache.org> on May 13, 2014, 2 pages	
	4	Artificial intelligence, retrieved from <URL: http://wikipedia.com> on May 21, 2014, 28 pages	
	5	Artificial neural network, retrieved from <URL: http://wikipedia.com> on May 21, 2014, 12 pages	
	6	Branch (computer science), retrieved from <URL: http://wikipedia.com> on May 6, 2014, 2 pages	
	7	Branch trace, retrieved from <URL: http://wikipedia.com> on May 28, 2014, 2 pages	
	8	C Function Call Conventions and the Stack, retrieved from <URL: http://www.csee.umbc.edu/~chang/cs313.s02/stack.shtml> on Jan 11, 2014, 4 pages	
	9	Call stack, retrieved from <URL: http://wikipedia.com> on Jan 11, 2014, 5 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	10	Program compilation and execution flow, retrieved from <URL: http://cs.stackexchange.com/questions/6187/program-compilation-and-execution-flow> on Jan 9, 2014, 2 pages	
	11	Control unit, retrieved from <URL: http://wikipedia.com> on May 25, 2014, 2 pages	
	12	Create dynamic applications with javax.tools, retrieved from <URL: http://www.ibm.com/developerworks/library/j-jcomp/> on May 13, 2014, 11 pages	
	13	Creating a Debugging and Profiling Agent with JVMTI, retrieved from <URL: http://www.oracle.com/technetwork/articles/java/jvmti-136367.html> on Jan 11, 2014, 18 pages	
	14	Decision tree learning, retrieved from <URL: http://wikipedia.com> on Jun 24, 2014, 5 pages	
	15	Dynamic Bytecode Instrumentation, retrieved from <URL: http://www.drdobbs.com/tools/dynamic-bytecode-instrumentation/184406433> on Apr 26, 2014, 6 pages	
	16	Dynamic programming language, retrieved from <URL: http://wikipedia.com> on May 7, 2014, 4 pages	
	17	Dynamic recompilation, retrieved from <URL: http://wikipedia.com> on May 19, 2014, 4 pages	
	18	eval(), retrieved from <URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/eval> on May 7, 2014, 13 pages	
	19	Function, retrieved from <URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Function> on May 7, 2014, 11 pages	
	20	FunctionEnter2 Function, retrieved from <URL: https://msdn.microsoft.com/en-us/library/aa964981(d=default,l=en-us,v=vs.110).aspx> on Mar 19, 2014, 2 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	21	FunctionLeave2 Function, retrieved from <URL: https://msdn.microsoft.com/en-us/library/aa964942(d=default,l=en-us,v=vs.110).aspx> on Mar 19, 2014, 1 pages	
	22	Heuristic, retrieved from <URL: http://wikipedia.com> on May 22, 2014, 5 pages	
	23	How to: Configure Trace Switches, retrieved from <URL: https://msdn.microsoft.com/en-us/library/vstudio/t06xyy08(d=default,l=en-us,v=vs.100).aspx> on Mar 19, 2014, 3 pages	
	24	Indirect branch, retrieved from <URL: http://wikipedia.com> on May 5, 2014, 1 pages	
	25	Instruction cycle, retrieved from <URL: http://wikipedia.com> on Apr 30, 2014, 2 pages	
	26	Instruction set, retrieved from <URL: http://wikipedia.com> on Apr 29, 2014, 6 pages	
	27	Interrupt, retrieved from <URL: http://wikipedia.com> on May 15, 2014, 5 pages	
	28	Introduction to Instrumentation and Tracing, retrieved from <URL: http://msdn.microsoft.com/en-us/library/aa983649(d=default,l=en-us,v=vs.71).aspx> on Jan 10, 2014, 2 pages	
	29	Java bytecode, retrieved from <URL: http://wikipedia.com> on May 27, 2014, 4 pages	
	30	Package java.util.logging, retrieved from <URL: http://docs.oracle.com/javase/1.5.0/docs/api/java/util/logging/package-summary.html> on Mar 19, 2014, 2 pages	
	31	javac - Java programming language compiler, retrieved from <URL: http://docs.oracle.com/javase/7/docs/technotes/tools/windows/javac.html> on May 13, 2014, 12 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	32	Interface JavaCompiler, retrieved from <URL: http://wikipedia.com> on May 13, 2014, 4 pages	
	33	Javassist, retrieved from <URL: http://wikipedia.com> on May 13, 2014, 2 pages	
	34	Just-in-time compilation, retrieved from <URL: http://wikipedia.com> on May 19, 2014, 5 pages	
	35	JVM Tool Interface, retrieved from <URL: http://docs.oracle.com/javase/7/docs/platform/jvmti/jvmti.html> on Apr 26, 2014, 127 pages	
	36	Machine code, retrieved from <URL: http://wikipedia.com> on Apr 30, 2014, 3 pages	
	37	Metaprogramming, retrieved from <URL: http://wikipedia.com> on May 8, 2014, 3 pages	
	38	Microsoft.VisualBasic.Logging Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/ms128122(d=default,l=en-us,v=vs.110).aspx> on Jan 13, 2014, 1 pages	
	39	Pin (computer program), retrieved from <URL: http://wikipedia.com> on May 6, 2014, 3 pages	
	40	Processor register, retrieved from <URL: http://wikipedia.com> on Apr 29, 2014, 4 pages	
	41	Profiling (computer programming), retrieved from <URL: http://wikipedia.com> on Jan 10, 2014, 4 pages	
	42	Profiling Overview, retrieved from <URL: http://msdn.microsoft.com/en-us/library/bb384493(d=default,l=en-us,v=vs.110).aspx> on Jan 12, 2014, 6 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number		16584736
	Filing Date		2019-09-26
	First Named Inventor	Jasmin Cosic	
	Art Unit		
	Examiner Name		
	Attorney Docket Number		

	43	Program counter, retrieved from <URL: http://wikipedia.com> on Apr 24, 2014, 2 pages	
	44	ptrace, retrieved from <URL: http://wikipedia.com> on Apr 24, 2014, 2 pages	
	45	Reflection (computer programming), retrieved from <URL: http://wikipedia.com> on May 7, 2014, 5 pages	
	46	Class Runtime, retrieved from <URL: http://docs.oracle.com/javase/7/docs/api/java/lang/Runtime.html> on Jan 10, 2014, 10 pages	
	47	Self-modifying code, retrieved from <URL: http://wikipedia.com> on Jan 12, 2014, 6 pages	
	48	Subjective logic, retrieved from <URL: http://wikipedia.com> on May 22, 2014, 5 pages	
	49	System.CodeDom.Compiler Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/z6b99ydt(d=default,l=en-us,v=vs.110).aspx> on May 13, 2014, 2 pages	
	50	System.Reflection.Emit Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/xd5fw18y(d=default,l=en-us,v=vs.110).aspx> on May 13, 2014, 2 pages	

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature		Date Considered	

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Jasmin Cosic/	Date (YYYY-MM-DD)	2019-09-29
Name/Print	Jasmin Cosic	Registration Number	

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

PTO/SB/08a (01-10)
Approved for use through 07/31/2012. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it contains a valid OMB control number.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS Remove

Examiner Initial*	Cite No	Patent Number	Kind Code[1]	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1					

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS Remove

Examiner Initial*	Cite No	Publication Number	Kind Code[1]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1					

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number[3]	Country Code[2] i	Kind Code[4]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T[5]
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T[5]

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	1	Tracing (software), retrieved from <URL: http://wikipedia.com> on Jan 10, 2014, 3 pages	
	2	Tree (data structure), retrieved from <URL: http://wikipedia.com> on Jun 24, 2014, 6 pages	
	3	PTRACE(2), retrieved from <URL: http://unixhelp.ed.ac.uk/CGI/man-cgi?ptrace> on Mar 19, 2014, 5 pages	
	4	Wevtutil, retrieved from <URL: http://technet.microsoft.com/en-us/library/cc732848(d=default,l=en-us,v=ws.11).aspx> on Apr 28, 2014, 5 pages	
	5	Intel Processor Trace, retrieved from <URL: https://software.intel.com/en-us/blogs/2013/09/18/processor-tracing> on Apr 28, 2014, 3 pages	
	6	YOUNGHOON JUNG, JAVA DYNAMICS Reflection and a lot more, Oct 10, 2012, 55 pages, Columbia University	
	7	AMITABH SRIVASTAVA, ALAN EUSTACE, ATOM A System for Building Customized Program Analysis Tools, May 3, 2004, 12 pages	
	8	MATHEW SMITHSON, KAPIL ANAND, APARNA KOTHA, KHALED ELWAZEER, NATHAN GILES, RAJEEV BARUA, Binary Rewriting without Relocation Information, Nov 10, 2010, 11 pages, University of Maryland	
	9	MAREK OLSZEWSKI, KEIR MIERTE, ADAM CZAJKOWSKI, ANGELA DEMLE BROWN, JIT Instrumentation - A Novel Approach To Dynamically Instrument Operating Systems, Feb 12, 2007, 14 pages, University of Toronto	

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature		Date Considered	

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Jasmin Cosic/	Date (YYYY-MM-DD)	2019-09-29
Name/Print	Jasmin Cosic	Registration Number	

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS Remove

Examiner Initial*	Cite No	Patent Number	Kind Code1	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1					

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS Remove

Examiner Initial*	Cite No	Publication Number	Kind Code1	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	20030026588		2003-02-06	Elder, James H. ; et al.	
	2	20040249774		2004-12-09	Caid, William R. ; et al.	
	3	20040267521		2004-12-30	Cutler, Ross ; et al.	
	4	20050245303		2005-11-03	Graepel, Kurt Hartwig ; et al.	
	5	20060047612		2006-03-02	Stanley; Kenneth O. ; et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	6	20070058856		2007-03-15	Boregowda; Lokesh R. ; et al.	
	7	20070106633		2007-05-10	Reiner; Bruce	
	8	20080144893		2008-06-19	GUO; Chun Biao ; et al.	
	9	20080288259		2008-11-20	Chambers; Robert L. ; et al.	
	10	20090110061		2009-04-30	LEE; Kang Yong ; et al.	
	11	20090131152		2009-05-21	Busse; Martin A.	
	12	20090141969		2009-06-04	Yu; Kai ; et al.	
	13	20090222388		2009-09-30	Hua; Wei ; et al.	
	14	20090324010		2009-12-31	Hou; Billy	
	15	20100033780		2010-02-11	Gitter; Kurt	
	16	20100063949		2010-03-11	EATON; JOHN ERIC ; et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	17	20100241595		2010-09-23	Felsher; David Paul	
	18	20100278420		2010-11-04	Shet; Vinay Damodar ; et al.	
	19	20110030031		2011-02-03	Lussier; Paul ; et al.	
	20	20130156345		2013-06-20	Shmunk; Dmitry Valerievich	
	21	20130159021		2013-06-20	Felsher; David Paul	
	22	20130278631		2013-10-24	Border; John N. ; et al.	
	23	20140052717		2014-02-20	FRANKS; Kasian ; et al.	
	24	20140161250		2014-06-12	Leeds; Richard ; et al.	
	25	20140177946		2014-06-26	LIM; Kil-Taek ; et al.	
	26	20140207580		2014-06-24	Minnis; William Charles ; et al.	
	27	20150039304		2015-02-05	Wein; Ron	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	28	20150055821		2015-02-26	Fotland; David Allen	

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number3	Country Code2 i	Kind Code4	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T5
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T5
	1	ABOUT | OpenCV, retrieved from <URL: http://opencv.org/about.html> on Dec 13, 2014, 1 pages	
	2	Animetrics, Inc. | 3D Facial Recognition, retrieved from <URL: http://animetrics.com/> on Dec 13, 2014, 2 pages	
	3	Bag-of-words model, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	4	Beier–Neely morphing algorithm, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 1 pages	
	5	Computer vision, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 7 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number		16584736
	Filing Date		2019-09-26
	First Named Inventor	Jasmin Cosic	
	Art Unit		
	Examiner Name		
	Attorney Docket Number		

	6	Convolutional neural network, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 5 pages	
	7	Decimation (signal processing), retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 3 pages	
	8	Digital image processing, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 3 pages	
	9	Dissolve (filmmaking), retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 2 pages	
	10	Facial recognition system, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 6 pages	
	11	Feature detection (computer vision), retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 3 pages	
	12	Feature extraction, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	13	Gesture recognition, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 5 pages	
	14	GrabCut, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 1 pages	
	15	Image processing, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	16	Image segmentation, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 12 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	17	Image warping, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 2 pages	
	18	Inbetweening, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 2 pages	
	19	Interpolation, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 5 pages	
	20	Language model, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 4 pages	
	21	List of speech recognition software, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 4 pages	
	22	Livewire Segmentation Technique, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 2 pages	
	23	Morphing, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 2 pages	
	24	Motion estimation, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	25	Muse: Face Morph || Mesh Warping, retrieved from <URL: http://alexwolfe.blogspot.com/2011/10/face-morph-mesh-warping.html> on Nov 19, 2015, 2 pages	
	26	n-gram, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 5 pages	
	27	Optical flow, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 4 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	
Application Number	16584736
Filing Date	2019-09-26
First Named Inventor	Jasmin Cosic
Art Unit	
Examiner Name	
Attorney Docket Number	

	28	Outline of object recognition, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 7 pages	
	29	Phonotactics, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 3 pages	
	30	Recurrent neural network, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 5 pages	
	31	Sample rate conversion, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 2 pages	
	32	Simple interactive object extraction, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	33	Speech recognition, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 12 pages	
	34	Speech segmentation, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 4 pages	
	35	Welcome to recognize-speech.com, retrieved from <URL: http://recognize-speech.com/> on Oct 18, 2015, 1 pages	
	36	Introduction Speech, retrieved from <URL: http://recognize-speech.com/speech> on Oct 18, 2015, 1 pages	
	37	Preprocessing, retrieved from <URL: http://recognize-speech.com/preprocessing> on Oct 18, 2015, 4 pages	
	38	Feature Extraction, retrieved from <URL: http://recognize-speech.com/feature-extraction> on Oct 18, 2015, 3 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	39	Acoustic model, retrieved from <URL: http://recognize-speech.com/acoustic-model> on Oct 18, 2015, 2 pages	
	40	Video content analysis, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	41	Video tracking, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 2 pages	
	42	Andrej Karpathy, Thomas Leung, George Toderici, Rahul Sukthankar, Sanketh Shetty, Li Fei-Fei, Large-scale Video Classification with Convolutional Neural Networks, Apr 14, 2014, 8 pages, Stanford University	
	43	Karen Simonyan, Andrew Zisserman, Two-Stream Convolutional Networks for Action Recognition in Videos, Nov 13, 2014, 11 pages, University of Oxford	

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature		Date Considered	

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Jasmin Cosic/	Date (YYYY-MM-DD)	2019-09-29
Name/Print	Jasmin Cosic	Registration Number	

This collection of information is required by 37 CFR 1.97 and 1.98. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 1 hour to complete, including gathering, preparing and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

PTO/SB/08a (01-10)
Approved for use through 07/31/2012. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS Remove

Examiner Initial*	Cite No	Patent Number	Kind Code1	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	7113946		2006-09-26	Cosic	
	2	7117225		2006-10-03	Cosic	
	3	8335805		2012-12-18	Cosic	
	4	8417740		2013-04-09	Cosic	
	5	8572035		2013-10-29	Cosic	
	6	8655900		2014-02-18	Cosic	
	7	9047324		2015-06-02	Cosic	
	8	9282309		2016-03-08	Cosic	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	9	9298749		2016-03-29	Cosic	
	10	9367806		2016-06-14	Cosic	
	11	9443192		2016-09-13	Cosic	
	12	9582762		2017-02-28	Cosic	
	13	9595294		2017-03-14	Cosic	
	14	9697869		2017-07-04	Cosic	

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS Remove

Examiner Initial*	Cite No	Publication Number	Kind Code1	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	20030065662		2003-04-03	Cosic	
	2	20040194017		2004-09-30	Cosic	
	3	20050149517		2005-07-07	Cosic	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	4	20050149542		2005-07-07	Cosic	
	5	20050289105		2005-12-29	Cosic	
	6	20100023541		2010-01-28	Cosic	
	7	20100082536		2010-04-01	Cosic	
	8	20130218932		2013-08-22	Cosic	
	9	20130226974		2013-08-29	Cosic	
	10	20160140999		2016-05-19	Cosic	
	11	20160142650		2016-05-19	Cosic	
	12	20160246819		2016-08-25	Cosic	
	13	20160246850		2016-08-25	Cosic	
	14	20160246868		2016-08-25	Cosic	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	15	20160292185		2016-10-06	Cosic	

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number[3]	Country Code[2] i	Kind Code[4]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T[5]
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T[5]
	1		

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature		Date Considered	

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Jasmin Cosic/	Date (YYYY-MM-DD)	2019-09-29
Name/Print	Jasmin Cosic	Registration Number	

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Electronic Acknowledgement Receipt

EFS ID:	37310485
Application Number:	16584736
International Application Number:	
Confirmation Number:	3341
Title of Invention:	MACHINE LEARNING FOR COMPUTING ENABLED SYSTEMS AND/OR DEVICES
First Named Inventor/Applicant Name:	Jasmin Cosic
Customer Number:	116094
Filer:	Jasmin Cosic
Filer Authorized By:	
Attorney Docket Number:	
Receipt Date:	29-SEP-2019
Filing Date:	
Time Stamp:	22:15:21
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	no

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1	Information Disclosure Statement (IDS) Form (SB08)	IDS_Part1.pdf	620588 7ba4d30046e1242dd11fc5f429b5561f737371cc	no	13

Warnings:

Information:

2	Information Disclosure Statement (IDS) Form (SB08)	IDS_Part2.pdf	614288 bf42b3e4f001a7b90576f325046df4ad6078895f	no	5

Warnings:

Information:

A U.S. Patent Number Citation or a U.S. Publication Number Citation is required in the Information Disclosure Statement (IDS) form for autoloading of data into USPTO systems. You may remove the form to add the required data in order to correct the Informational Message if you are citing U.S. References. If you chose not to include U.S. References, the image of the form will be processed and be made available within the Image File Wrapper (IFW) system. However, no data will be extracted from this form. Any additional data such as Foreign Patent Documents or Non Patent Literature will be manually reviewed and keyed into USPTO systems.

3	Information Disclosure Statement (IDS) Form (SB08)	IDS_Part3.pdf	618053 8cdabce4bf5afea206f611ab56dc91f9f90e1b84	no	10

Warnings:

Information:

4	Information Disclosure Statement (IDS) Form (SB08)	IDS_Part4.pdf	615056 fcbecb7151d73748ba03899976ef3c73d5dd72c7	no	6

Warnings:

Information:

5	Non Patent Literature	NPL_Part1.pdf	19783223 81a3858c2f8d004a3e30bbae7ec3bfb2087ac1b9	no	380

Warnings:

The page size in the PDF is too large. The pages should be 8.5 x 11 or A4. If this PDF is submitted, the pages will be resized upon entry into the Image File Wrapper and may affect subsequent processing

Information:

6	Non Patent Literature	NPL_Part2.pdf	5883714 fac28c7df69b33d27768405ac8c3f0bb9481d1e4	no	114

Warnings:

The page size in the PDF is too large. The pages should be 8.5 x 11 or A4. If this PDF is submitted, the pages will be resized upon entry into the Image File Wrapper and may affect subsequent processing

Information:

7	Non Patent Literature	NPL_Part3.pdf	16903333 327e784f4ab311324ac99b3c6dd8b06ae0b170c7	no	160

Warnings:

Information:

Total Files Size (in bytes):	45038255

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.
National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.
New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

PATENT APPLICATION FEE DETERMINATION RECORD

Substitute for Form PTO-875

Application or Docket Number: 16/584,736

APPLICATION AS FILED - PART I

FOR	NUMBER FILED (Column 1)	NUMBER EXTRA (Column 2)	SMALL ENTITY RATE($)	SMALL ENTITY FEE($)	OR	OTHER THAN SMALL ENTITY RATE($)	OTHER THAN SMALL ENTITY FEE($)
BASIC FEE (37 CFR 1.16(a), (b), or (c))	N/A	N/A	N/A	75		N/A	
SEARCH FEE (37 CFR 1.16(k), (i), or (m))	N/A	N/A	N/A	330		N/A	
EXAMINATION FEE (37 CFR 1.16(o), (p), or (q))	N/A	N/A	N/A	380		N/A	
TOTAL CLAIMS (37 CFR 1.16(i))	20 minus 20 =	*	x 50 =	0.00	OR		
INDEPENDENT CLAIMS (37 CFR 1.16(h))	3 minus 3 =	*	x 230 =	0.00			
APPLICATION SIZE FEE (37 CFR 1.16(s))	If the specification and drawings exceed 100 sheets of paper, the application size fee due is $310 ($155 for small entity) for each additional 50 sheets or fraction thereof. See 35 U.S.C. 41(a)(1)(G) and 37 CFR 1.16(s).			200			
MULTIPLE DEPENDENT CLAIM PRESENT (37 CFR 1.16(j))				0.00			
* If the difference in column 1 is less than zero, enter "0" in column 2.			TOTAL	985		TOTAL	

APPLICATION AS AMENDED - PART II

AMENDMENT A

	CLAIMS REMAINING AFTER AMENDMENT (Column 1)		HIGHEST NUMBER PREVIOUSLY PAID FOR (Column 2)	PRESENT EXTRA (Column 3)	SMALL ENTITY RATE($)	SMALL ENTITY ADDITIONAL FEE($)	OR	OTHER THAN SMALL ENTITY RATE($)	OTHER THAN SMALL ENTITY ADDITIONAL FEE($)
Total (37 CFR 1.16(i))	*	Minus	**	=	x =		OR	x =	
Independent (37 CFR 1.16(h))	*	Minus	***	=	x =		OR	x =	
Application Size Fee (37 CFR 1.16(s))									
FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))							OR		
					TOTAL ADD'L FEE		OR	TOTAL ADD'L FEE	

AMENDMENT B

	CLAIMS REMAINING AFTER AMENDMENT (Column 1)		HIGHEST NUMBER PREVIOUSLY PAID FOR (Column 2)	PRESENT EXTRA (Column 3)	SMALL ENTITY RATE($)	SMALL ENTITY ADDITIONAL FEE($)	OR	OTHER THAN SMALL ENTITY RATE($)	OTHER THAN SMALL ENTITY ADDITIONAL FEE($)
Total (37 CFR 1.16(i))	*	Minus	**	=	x =		OR	x =	
Independent (37 CFR 1.16(h))	*	Minus	***	=	x =		OR	x =	
Application Size Fee (37 CFR 1.16(s))									
FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))							OR		
					TOTAL ADD'L FEE		OR	TOTAL ADD'L FEE	

* If the entry in column 1 is less than the entry in column 2, write "0" in column 3.
** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, enter "20".
*** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, enter "3".
The "Highest Number Previously Paid For" (Total or Independent) is the highest found in the appropriate box in column 1.

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NUMBER	FILING or 371(c) DATE	GRP ART UNIT	FIL FEE REC'D	ATTY.DOCKET.NO	TOT CLAIMS	IND CLAIMS
16/584,736	09/26/2019	2129	985		20	3

CONFIRMATION NO. 3341

FILING RECEIPT

OC000000111792270

116094
Jasmin Cosic
108 Woodbury Street
Pawtucket, RI 02861

Date Mailed: 10/11/2019

Receipt is acknowledged of this non-provisional utility patent application. The application will be taken up for examination in due course. Applicant will be notified as to the results of the examination. Any correspondence concerning the application must include the following identification information: the U.S. APPLICATION NUMBER, FILING DATE, NAME OF FIRST INVENTOR, and TITLE OF INVENTION. Fees transmitted by check or draft are subject to collection.

Please verify the accuracy of the data presented on this receipt. If an error is noted on this Filing Receipt, please submit a written request for a corrected Filing Receipt, including a properly marked-up ADS showing the changes with strike-through for deletions and underlining for additions. If you received a "Notice to File Missing Parts" or other Notice requiring a response for this application, please submit any request for correction to this Filing Receipt with your reply to the Notice. When the USPTO processes the reply to the Notice, the USPTO will generate another Filing Receipt incorporating the requested corrections provided that the request is grantable.

Inventor(s)

Jasmin Cosic, Miami, FL;

Applicant(s)

Jasmin Cosic, Miami, FL;

Power of Attorney: None

Domestic Priority data as claimed by applicant

This application is a CON of 15/822,150 11/26/2017

Foreign Applications for which priority is claimed (You may be eligible to benefit from the **Patent Prosecution Highway** program at the USPTO. Please see http://www.uspto.gov for more information.) - None.
Foreign application information must be provided in an Application Data Sheet in order to constitute a claim to foreign priority. See 37 CFR 1.55 and 1.76.

Permission to Access Application via Priority Document Exchange: No

Permission to Access Search Results: No

Applicant may provide or rescind an authorization for access using Form PTO/SB/39 or Form PTO/SB/69 as appropriate.

If Required, Foreign Filing License Granted: 10/09/2019

The country code and number of your priority application, to be used for filing abroad under the Paris Convention, is **US 16/584,736**

Projected Publication Date: Request for Non-Publication Acknowledged

Non-Publication Request: Yes

Early Publication Request: No
**** SMALL ENTITY ****
Title

MACHINE LEARNING FOR COMPUTING ENABLED SYSTEMS AND/OR DEVICES

Preliminary Class

706

Statement under 37 CFR 1.55 or 1.78 for AIA (First Inventor to File) Transition Applications: No

PROTECTING YOUR INVENTION OUTSIDE THE UNITED STATES

Since the rights granted by a U.S. patent extend only throughout the territory of the United States and have no effect in a foreign country, an inventor who wishes patent protection in another country must apply for a patent in a specific country or in regional patent offices. Applicants may wish to consider the filing of an international application under the Patent Cooperation Treaty (PCT). An international (PCT) application generally has the same effect as a regular national patent application in each PCT-member country. The PCT process **simplifies** the filing of patent applications on the same invention in member countries, but **does not result** in a grant of "an international patent" and does not eliminate the need of applicants to file additional documents and fees in countries where patent protection is desired.

Almost every country has its own patent law, and a person desiring a patent in a particular country must make an application for patent in that country in accordance with its particular laws. Since the laws of many countries differ in various respects from the patent law of the United States, applicants are advised to seek guidance from specific foreign countries to ensure that patent rights are not lost prematurely.

Applicants also are advised that in the case of inventions made in the United States, the Director of the USPTO must issue a license before applicants can apply for a patent in a foreign country. The filing of a U.S. patent application serves as a request for a foreign filing license. The application's filing receipt contains further information and guidance as to the status of applicant's license for foreign filing.

Applicants may wish to consult the USPTO booklet, "General Information Concerning Patents" (specifically, the section entitled "Treaties and Foreign Patents") for more information on timeframes and deadlines for filing foreign patent applications. The guide is available either by contacting the USPTO Contact Center at 800-786-9199, or it can be viewed on the USPTO website at http://www.uspto.gov/web/offices/pac/doc/general/index.html.

For information on preventing theft of your intellectual property (patents, trademarks and copyrights), you may wish to consult the U.S. Government website, http://www.stopfakes.gov. Part of a Department of Commerce initiative, this website includes self-help "toolkits" giving innovators guidance on how to protect intellectual property in specific countries such as China, Korea and Mexico. For questions regarding patent enforcement issues, applicants may call the U.S. Government hotline at 1-866-999-HALT (1-866-999-4258).

LICENSE FOR FOREIGN FILING UNDER

Title 35, United States Code, Section 184

Title 37, Code of Federal Regulations, 5.11 & 5.15

GRANTED

The applicant has been granted a license under 35 U.S.C. 184, if the phrase "IF REQUIRED, FOREIGN FILING LICENSE GRANTED" followed by a date appears on this form. Such licenses are issued in all applications where the conditions for issuance of a license have been met, regardless of whether or not a license may be required as set forth in 37 CFR 5.15. The scope and limitations of this license are set forth in 37 CFR 5.15(a) unless an earlier license has been issued under 37 CFR 5.15(b). The license is subject to revocation upon written notification. The date indicated is the effective date of the license, unless an earlier license of similar scope has been granted under 37 CFR 5.13 or 5.14.

This license is to be retained by the licensee and may be used at any time on or after the effective date thereof unless it is revoked. This license is automatically transferred to any related applications(s) filed under 37 CFR 1.53(d). This license is not retroactive.

The grant of a license does not in any way lessen the responsibility of a licensee for the security of the subject matter as imposed by any Government contract or the provisions of existing laws relating to espionage and the national security or the export of technical data. Licensees should apprise themselves of current regulations especially with respect to certain countries, of other agencies, particularly the Office of Defense Trade Controls, Department of State (with respect to Arms, Munitions and Implements of War (22 CFR 121-128)); the Bureau of Industry and Security, Department of Commerce (15 CFR parts 730-774); the Office of Foreign AssetsControl, Department of Treasury (31 CFR Parts 500+) and the Department of Energy.

NOT GRANTED

No license under 35 U.S.C. 184 has been granted at this time, if the phrase "IF REQUIRED, FOREIGN FILING LICENSE GRANTED" DOES NOT appear on this form. Applicant may still petition for a license under 37 CFR 5.12, if a license is desired before the expiration of 6 months from the filing date of the application. If 6 months has lapsed from the filing date of this application and the licensee has not received any indication of a secrecy order under 35 U.S.C. 181, the licensee may foreign file the application pursuant to 37 CFR 5.15(b).

SelectUSA

The United States represents the largest, most dynamic marketplace in the world and is an unparalleled location for business investment, innovation, and commercialization of new technologies. The U.S. offers tremendous resources and advantages for those who invest and manufacture goods here. Through SelectUSA, our nation works to promote and facilitate business investment. SelectUSA provides information assistance to the international investor community; serves as an ombudsman for existing and potential investors; advocates on behalf of U.S. cities, states, and regions competing for global investment; and counsels U.S. economic development organizations on investment attraction best practices. To learn more about why the United States is the best country in the world to develop technology, manufacture products, deliver services, and grow your business, visit http://www.SelectUSA.gov or call +1-202-482-6800.

PTO/SB/06 (09-11)
Approved for use through 1/31/2014. OMB 0651-0032
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

PATENT APPLICATION FEE DETERMINATION RECORD Substitute for Form PTO-875	Application or Docket Number 16/584,736	Filing Date 09/26/2019	☐ To be Mailed

ENTITY: ☐ LARGE ☑ SMALL ☐ MICRO

APPLICATION AS FILED - PART I

FOR	(Column 1) NUMBER FILED	(Column 2) NUMBER EXTRA		RATE ($)	FEE ($)
☐ BASIC FEE (37 CFR 1.16(a), (b), or (c))	N/A	N/A		N/A	
☐ SEARCH FEE (37 CFR 1.16(k), (i), or (m))	N/A	N/A		N/A	
☐ EXAMINATION FEE (37 CFR 1.16(o), (p), or (q))	N/A	N/A		N/A	
TOTAL CLAIMS (37 CFR 1.16(i))	minus 20 =	*		x $50 =	
INDEPENDENT CLAIMS (37 CFR 1.16(h))	minus 3 =	*		x $230 =	
☐ APPLICATION SIZE FEE (37 CFR 1.16(s))	If the specification and drawings exceed 100 sheets of paper, the application size fee due is $310 ($155 for small entity) for each additional 50 sheets or fraction thereof. See 35 U.S.C. 41(a)(1)(G) and 37 CFR 1.16(s).				
☐ MULTIPLE DEPENDENT CLAIM PRESENT (37 CFR 1.16(j))					
* If the difference in column 1 is less than zero, enter "0" in column 2.				TOTAL	

APPLICATION AS AMENDED - PART II

		(Column 1)		(Column 2)	(Column 3)			
AMENDMENT	12/26/2020	CLAIMS REMAINING AFTER AMENDMENT		HIGHEST NUMBER PREVIOUSLY PAID FOR	PRESENT EXTRA		RATE ($)	ADDITIONAL FEE ($)
	Total (37 CFR 1.16(i))	* 20	Minus	** 20	= 0		x $50 =	0
	Independent (37 CFR 1.16(h))	* 3	Minus	*** 3	= 0		x $240 =	0
	☐ Application Size Fee (37 CFR 1.16(s))							
	☐ FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))							
							TOTAL ADD'L FEE	0

		(Column 1)		(Column 2)	(Column 3)			
AMENDMENT		CLAIMS REMAINING AFTER AMENDMENT		HIGHEST NUMBER PREVIOUSLY PAID FOR	PRESENT EXTRA		RATE ($)	ADDITIONAL FEE ($)
	Total (37 CFR 1.16(i))	*	Minus	**	=		x $0 =	
	Independent (37 CFR 1.16(h))	*	Minus	***	=		x $0 =	
	☐ Application Size Fee (37 CFR 1.16(s))							
	☐ FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))							
							TOTAL ADD'L FEE	

* If the entry in column 1 is less than the entry in column 2, write "0" in column 3.	LDRC
** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, enter "20".	/EVA V GILLIS/
*** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, enter "3".	
The "Highest Number Previously Paid For" (Total or Independent) is the highest number found in the appropriate box in column 1.	

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Inventor:	Jasmin Cosic	Confirmation No.: 3341
Title:	"MACHINE LEARNING FOR COMPUTING ENABLED SYSTEMS AND/OR DEVICES"	
Serial No.:	16/584,736	Filed: September 26, 2019
Examiner:	JOHNS, ANDREW W	Group Art Unit: 2665

Via EFS-Web December 26, 2020

Mail Stop Amendment
COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, VA 22313-1450

PRELIMINARY AMENDMENT

Dear Commissioner:

Before examination on the merits, please amend the claims in the above-identified application as shown in the following listing of claims.

Inventor: Jasmin Cosic
Serial No.: 16/584,736
Filing Date: September 26, 2019

Listing of Claims

1 - 20 (canceled)

21. (new) A non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising:

receiving a first digital picture from a picture capturing apparatus;

receiving a first one or more instruction sets for operating a device; and

learning the first digital picture correlated with the first one or more instruction sets for operating the device.

22. (new) The non-transitory computer storage medium of claim 21, wherein the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from an application for operating the device using at least a tracing of the application for operating the device, wherein the application for operating the device runs on the one or more processor circuits or on another one or more processor circuits.

23. (new) The non-transitory computer storage medium of claim 21, wherein the receiving the first one or more instruction sets for operating the device includes

extracting the first one or more instruction sets for operating the device from at least one portion of the one or more processor circuits or from at least one portion of another one or more processor circuits.

24. (new) The non-transitory computer storage medium of claim 21, wherein the first digital picture includes a still digital picture or a motion digital picture.

25. (new) A method comprising:

(a) receiving a first digital picture from a picture capturing apparatus, the receiving of (a) performed by one or more processor circuits;

(b) receiving a first one or more instruction sets for operating a device, the receiving of (b) performed by the one or more processor circuits; and

(c) learning the first digital picture correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the one or more processor circuits.

26. (new) The method of claim 25, wherein the receiving of (b) includes receiving the first one or more instruction sets for operating the device from an application for operating the device using at least a tracing of the application for operating the device, wherein the application for operating the device runs on the one or more processor circuits or on another one or more processor circuits.

27. (new) The method of claim 25, wherein the receiving of (b) includes extracting the first one or more instruction sets for operating the device from at least one portion of the one or more processor circuits or from at least one portion of another one or more processor circuits.

28. (new) The method of claim 25, wherein the first digital picture includes a still digital picture or a motion digital picture.

29. (new) A system comprising:

one or more processor circuits configured to execute instruction sets for operating a device;

a memory unit configured to store data;

a picture capturing apparatus configured to capture digital pictures; and

a learning unit configured to:

receive a first digital picture from the picture capturing apparatus;

receive a first one or more instruction sets for operating the device; and

learn the first digital picture correlated with the first one or more instruction sets for operating the device.

30. (new) The system of claim 29, wherein the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from an application for operating the device using at least a tracing of the application for operating the device, wherein

the application for operating the device runs on the one or more processor circuits or on another one or more processor circuits.

31. (new) The system of claim 29, wherein the receiving the first one or more instruction sets for operating the device includes extracting the first one or more instruction sets for operating the device from at least one portion of the one or more processor circuits or from at least one portion of another one or more processor circuits.

32. (new) The system of claim 29, wherein the learning the first digital picture correlated with the first one or more instruction sets for operating the device includes correlating the first digital picture with the first one or more instruction sets for operating the device.

33. (new) The system of claim 29, wherein the learning the first digital picture correlated with the first one or more instruction sets for operating the device includes causing the memory unit to store the first digital picture correlated with the first one or more instruction sets for operating the device, and wherein the memory unit stores at least a plurality of digital pictures correlated with one or more instruction sets for operating the device.

34. (new) The system of claim 29, wherein the first digital picture correlated with the first one or more instruction sets for operating the device include a unit of knowledge of how the device operated in a visual surrounding.

35. (new) The system of claim 29, wherein the first digital picture includes a still digital picture or a motion digital picture.

36. (new) The system of claim 29, wherein the learning unit includes an interface for attaching to at least one of: the device, the one or more processor circuits, another one or more processor circuits, or an application for operating the device, wherein the application for operating the device runs on the one or more processor circuits or on another one or more processor circuits.

37. (new) The system of claim 29, wherein the one or more processor circuits include one or more microcontrollers.

38. (new) The system of claim 29, wherein at least one of: the one or more processor circuits or the picture capturing apparatus are included in a first computing device and at least one of: the memory unit or the learning unit are included in second computing device.

39. (new) The system of claim 29, wherein at least one of: the one or more processor circuits, the memory unit, the picture capturing apparatus, or the learning unit are included in the device.

40. (new) The system of claim 29, wherein the learning unit includes at least one of: a hardware element that is included in the one or more processor circuits, a hardware element that is included in another one or more processor circuits, an application operating on the one or more processor circuits, an application operating on another one or more processor circuits, or an element coupled to the one or more processor circuits.

Remarks

In this preliminary amendment, the applicant cancels claims 1-20, and presents for examination new claims 21-40. After entry of this preliminary amendment, claims 21-40 are pending. The undersigned may be contacted at (317) 772-1312 concerning this application.

I hereby certify that this correspondence is being submitted electronically via EFS Web to the United States Patent and Trademark Office.

By /Jasmin Cosic/
Jasmin Cosic

Date submitted: December 26, 2020

Respectfully submitted,

/Jasmin Cosic/

Jasmin Cosic

Electronic Acknowledgement Receipt

EFS ID:	41490785
Application Number:	16584736
International Application Number:	
Confirmation Number:	3341
Title of Invention:	MACHINE LEARNING FOR COMPUTING ENABLED SYSTEMS AND/OR DEVICES
First Named Inventor/Applicant Name:	Jasmin Cosic
Customer Number:	116094
Filer:	Jasmin Cosic
Filer Authorized By:	
Attorney Docket Number:	
Receipt Date:	26-DEC-2020
Filing Date:	26-SEP-2019
Time Stamp:	15:38:21
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	no

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1		PRELIMINARY_AMENDMENT.pdf	32364 c8e5c17b5652dbd66eee23ebd13c543c104dfeec	yes	8

	Multipart Description/PDF files in .zip description		
	Document Description	**Start**	**End**
	Preliminary Amendment	1	1
	Claims	2	7
	Applicant Arguments/Remarks Made in an Amendment	8	8

Warnings:

Information:

Total Files Size (in bytes):	32364

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.
National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.
New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
16/584,736	09/26/2019	Jasmin Cosic		3341

116094 7590 02/01/2021
Jasmin Cosic
108 Woodbury Street
Pawtucket, RI 02861

EXAMINER	
JOHNS, ANDREW W	

ART UNIT	PAPER NUMBER
2665	

MAIL DATE	DELIVERY MODE
02/01/2021	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	**Application No.** 16/584,736	**Applicant(s)** Cosic, Jasmin	
	Examiner ANDREW W JOHNS	**Art Unit** 2665	**AIA (FITF) Status** Yes

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTHS FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133).
  Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☑ Responsive to communication(s) filed on 26 December 2020.
☐ A declaration(s)/affidavit(s) under **37 CFR 1.130(b)** was/were filed on _____.

2a)☐ This action is **FINAL.** 2b) ☑ This action is non-final.

3)☐ An election was made by the applicant in response to a restriction requirement set forth during the interview on _____; the restriction requirement and election have been incorporated into this action.

4)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims*

5) ☑ Claim(s) 21-40 is/are pending in the application.
5a) Of the above claim(s) _____ is/are withdrawn from consideration.

6) ☐ Claim(s) _____ is/are allowed.

7) ☑ Claim(s) 21-40 is/are rejected.

8) ☐ Claim(s) _____ is/are objected to.

9) ☐ Claim(s) _____ are subject to restriction and/or election requirement

* If any claims have been determined allowable, you may be eligible to benefit from the **Patent Prosecution Highway** program at a participating intellectual property office for the corresponding application. For more information, please see http://www.uspto.gov/patents/init_events/pph/index.jsp or send an inquiry to **PPHfeedback@uspto.gov.**

Application Papers

10)☐ The specification is objected to by the Examiner.

11)☑ The drawing(s) filed on 26 September 2019 is/are: a)☑ accepted or b)☐ objected to by the Examiner.
Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).
Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
Certified copies:
a)☐ All b)☐ Some** c)☐ None of the:
1.☐ Certified copies of the priority documents have been received.
2.☐ Certified copies of the priority documents have been received in Application No. _____.
3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

** See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☑ Notice of References Cited (PTO-892)

2) ☑ Information Disclosure Statement(s) (PTO/SB/08a and/or PTO/SB/08b)
Paper No(s)/Mail Date ______.

3) ☐ Interview Summary (PTO-413)
Paper No(s)/Mail Date ______.

4) ☐ Other: ______.

DETAILED ACTION

Notice of Pre-AIA or AIA Status

1. The present application, filed on or after March 16, 2013, is being examined under the first inventor to file provisions of the AIA.

Specification

2. The lengthy specification has not been checked to the extent necessary to determine the presence of all possible minor errors. Applicant's cooperation is requested in correcting any errors of which applicant may become aware in the specification.

Claim Interpretation

3. The following is a quotation of 35 U.S.C. § 112(f):

> (f) Element in Claim for a Combination. – An element in a claim for a combination may be expressed as a means or step for performing a specified function without the recital of structure, material, or acts in support thereof, and such claim shall be construed to cover the corresponding structure, material, or acts described in the specification and equivalents thereof.

The following is a quotation of pre-AIA 35 U.S.C. § 112, sixth paragraph:

> An element in a claim for a combination may be expressed as a means or step for performing a specified function without the recital of structure, material, or acts in support thereof, and such claim shall be construed to cover the corresponding structure, material, or acts described in the specification and equivalents thereof.

4. The claims in this application are given their broadest reasonable interpretation using the plain meaning of the claim language in light of the specification as it would be understood by one of ordinary skill in the art. The broadest reasonable interpretation of a claim element (also commonly referred to as a claim limitation) is limited by the description in the specification when 35 U.S.C. § 112(f) or pre-AIA 35 U.S.C. § 112, sixth paragraph, is invoked.

As explained in M.P.E.P. § 2181, subsection I, claim limitations that meet the following three-prong test will be interpreted under 35 U.S.C. § 112(f) or pre-AIA 35 U.S.C. § 112, sixth paragraph:

(A) the claim limitation uses the term "means" or "step" or a term used as a substitute for "means" that is a generic placeholder (also called a nonce term or a non-structural term having no specific structural meaning) for performing the claimed function;

(B) the term "means" or "step" or the generic placeholder is modified by functional language, typically, but not always linked by the transition word "for" (e.g., "means for") or another linking word or phrase, such as "configured to" or "so that"; and

(C) the term "means" or "step" or the generic placeholder is not modified by sufficient structure, material, or acts for performing the claimed function.

Use of the word "means" (or "step") in a claim with functional language creates a rebuttable presumption that the claim limitation is to be treated in accordance with 35 U.S.C. § 112(f) or pre-AIA 35 U.S.C. § 112, sixth paragraph. The presumption that the claim limitation is interpreted under 35 U.S.C. § 112(f) or pre-AIA 35 U.S.C. § 112, sixth paragraph, is rebutted when the claim limitation recites sufficient structure, material, or acts to entirely perform the recited function.

Absence of the word "means" (or "step") in a claim creates a rebuttable presumption that the claim limitation is not to be treated in accordance with 35 U.S.C. § 112(f) or pre-AIA 35 U.S.C. § 112, sixth paragraph. The presumption that the claim limitation is not interpreted under 35 U.S.C. § 112(f) or pre-AIA 35 U.S.C. § 112, sixth paragraph, is rebutted when the

claim limitation recites function without reciting sufficient structure, material or acts to entirely perform the recited function.

Claim limitations in this application that use the word "means" (or "step") are being interpreted under 35 U.S.C. § 112(f) or pre-AIA 35 U.S.C. § 112, sixth paragraph, except as otherwise indicated in an Office action. Conversely, claim limitations in this application that do not use the word "means" (or "step") are not being interpreted under 35 U.S.C. § 112(f) or pre-AIA 35 U.S.C. § 112, sixth paragraph, except as otherwise indicated in an Office action.

5. This application includes one or more claim limitations that do not use the word "means," but are nonetheless being interpreted under 35 U.S.C. § 112(f) or pre-AIA 35 U.S.C. § 112, sixth paragraph, because the claim limitation(s) uses a generic placeholder that is coupled with functional language without reciting sufficient structure to perform the recited function and the generic placeholder is not preceded by a structural modifier. Such claim limitation(s) is/are: "memory unit", "picture capturing apparatus", and "learning unit" in claim 29.[1]

Because this/these claim limitation(s) is/are being interpreted under 35 U.S.C. § 112(f) or pre-AIA 35 U.S.C. § 112, sixth paragraph, it/they is/are being interpreted to cover the corresponding structure described in the specification as performing the claimed function, and equivalents thereof.

If applicant does not intend to have this/these limitation(s) interpreted under 35 U.S.C. § 112(f) or pre-AIA 35 U.S.C. § 112, sixth paragraph, applicant may: (1) amend the claim

---

[1] Note that dependent claim 40 further defines the "learning unit" as including sufficient structure to accomplish the claimed functionality, so that the "learning unit" is NOT interpreted under 35 USC 112(f) or pre-AIA 35 USC 112, sixth paragraph in dependent claim 40.

limitation(s) to avoid it/them being interpreted under 35 U.S.C. § 112(f) or pre-AIA 35 U.S.C. § 112, sixth paragraph (e.g., by reciting sufficient structure to perform the claimed function); or (2) present a sufficient showing that the claim limitation(s) recite(s) sufficient structure to perform the claimed function so as to avoid it/them being interpreted under 35 U.S.C. § 112(f) or pre-AIA 35 U.S.C. § 112, sixth paragraph.

Double Patenting

6. A rejection based on double patenting of the "same invention" type finds its support in the language of 35 U.S.C. § 101 which states that "whoever invents or discovers any new and useful process... may obtain a patent therefor..." (Emphasis added). Thus, the term "same invention," in this context, means an invention drawn to identical subject matter. See *Miller v. Eagle Mfg. Co.*, 151 U.S. 186 (1894); *In re Vogel*, 422 F.2d 438, 164 USPQ 619 (CCPA 1970); *In re Ockert*, 245 F.2d 467, 114 USPQ 330 (CCPA 1957).

A statutory type (35 U.S.C. § 101) double patenting rejection can be overcome by canceling or amending the claims that are directed to the same invention so they are no longer coextensive in scope. The filing of a terminal disclaimer cannot overcome a double patenting rejection based upon 35 U.S.C. § 101.

7. Claims 21-40 are rejected under 35 U.S.C. § 101 as claiming the same invention as that of claims 1-20 of prior U.S. Patent No. 10,474,934. This is a statutory double patenting rejection. Claims 21-40 are word-for-word identical to corresponding claims 1-20 of the '934 patent, so that they are clearly claiming identical subject matter.

Conclusion

8. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure. Perez et al. use learning to correlated a user's behavior with contextual circumstances. Williams et al. use images in a learning system. Tanigawa et al. teach learning correlations between information and received digital image data.

9. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Andrew Johns whose telephone number is (571) 272-7391. The examiner in normally available Monday through Friday, typically between 6:15 am and 2:45 pm Eastern Time. The examiner may also be contacted by e-mail using the address: andrew.johns@uspto.gov. (Applicant is reminded of the Office policy regarding e-mail communications. See M.P.E.P. § 502.03)

If attempts to reach the examiner are unsuccessful, the examiner's supervisor, Edward Urban, can be reached at (571) 272-7899. The fax phone number for this art unit is (571) 273-8300. In order to ensure prompt delivery to the examiner, all unofficial communications should be clearly labeled as "Draft" or "Unofficial."

A. Johns
29 January 2021

/Andrew W Johns/
Primary Examiner, Art Unit 2665

Notice of References Cited	Application/Control No. 16/584,736	Applicant(s)/Patent Under Reexamination Cosic, Jasmin	
	Examiner ANDREW W JOHNS	Art Unit 2665	Page 1 of 1

U.S. PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Name	CPC Classification	US Classification
*	A	US-2011/0007079-A1	01-2011	Perez et al.	H04N21/4532	345/473
*	B	US-9,488,984-B1	11-2016	Williams et al.	G06K9/2036	1/1
*	C	US-2018/0068564-A1	03-2018	TANIGAWA et al.	G08G1/143	1/1
*	D	US-10,474,934-B1	11-2019	Cosic	G06K9/6262	1/1
	E					
	F					
	G					
	H					
	I					
	J					
	K					
	L					
	M					

FOREIGN PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Country	Name	CPC Classification
	N					
	O					
	P					
	Q					
	R					
	S					
	T					

NON-PATENT DOCUMENTS

*		Include as applicable: Author, Title Date, Publisher, Edition or Volume, Pertinent Pages)
	U	
	V	
	W	
	X	

*A copy of this reference is not being furnished with this Office action. (See MPEP § 707.05(a).)
Dates in MM-YYYY format are publication dates. Classifications may be US or foreign.

Index of Claims	Application/Control No. 16/584,736	Applicant(s)/Patent Under Reexamination Cosic, Jasmin
	Examiner ANDREW W JOHNS	Art Unit 2665

✓	Rejected	-	Cancelled	N	Non-Elected	A	Appeal
=	Allowed	÷	Restricted	I	Interference	O	Objected

CLAIMS

☐ Claims renumbered in the same order as presented by applicant ☐ CPA ☐ T.D. ☐ R.1.47

CLAIM		DATE								
Final	Original	01/29/2021								
	21	✓								
	22	✓								
	23	✓								
	24	✓								
	25	✓								
	26	✓								
	27	✓								
	28	✓								
	29	✓								
	30	✓								
	31	✓								
	32	✓								
	33	✓								
	34	✓								
	35	✓								
	36	✓								
	37	✓								
	38	✓								
	39	✓								
	40	✓								

Search Notes	Application/Control No. 16/584,736	Applicant(s)/Patent Under Reexamination Cosic, Jasmin
	Examiner ANDREW W JOHNS	Art Unit 2665

CPC - Searched*		
Symbol	**Date**	**Examiner**
G06K 9/00355, 9/00389, 9/66	01/29/2021	/AWJ/
G06T 2207/20081	01/29/2021	/AWJ/
G06F 3/002, 3/005, 3/017	01/29/2021	/AWJ/
G06N 5/022, 5/025, 20/00	01/29/2021	/AWJ/

CPC Combination Sets - Searched*		
Symbol	**Date**	**Examiner**

US Classification - Searched*			
Class	**Subclass**	**Date**	**Examiner**

* See search history printout included with this form or the SEARCH NOTES box below to determine the scope of the search.

Search Notes		
Search Notes	**Date**	**Examiner**
Reviewed file history and prior art in parent application SN 15/822,150	01/28/2021	/AWJ/
Inventor name search in EAST (search history attached)	01/29/2021	/AWJ/

Interference Search			
US Class/CPC Symbol	**US Subclass/CPC Group**	**Date**	**Examiner**

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS Remove USPC/CPC

Examiner Initial*	Cite No	Patent Number	Kind Code1	Issue Date	Name of Patentee or Applicant of cited Document	~~Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear~~
/AWJ/	1	4370707	A	1983-01-25	Phillips , et al.	364/200
/AWJ/	2	4730315	A	1988-03-08	Saito , et al.	371/19
/AWJ/	3	4860203	A	1989-08-22	Corrigan , et al.	364/300
/AWJ/	4	5602982	A	1997-02-11	Judd , et al.	395/326
/AWJ/	5	6026234	A	2000-02-15	Hanson , et al.	395/704
/AWJ/	6	6088731	A	2000-07-11	Kiraly , et al.	709/229
/AWJ/	7	6106299	A	2000-08-22	Ackermann , et al.	434/81
/AWJ/	8	6126330	A	2000-10-03	Knight	395/704

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

/AWJ/	9	6314558	B1	2001-11-06	Angel , et al.	717/4
/AWJ/	10	6643842	B2	2003-11-04	Angel , et al.	717/130
/AWJ/	11	6728689	B1	2004-04-27	Drissi , et al.	706/14
/AWJ/	12	6735632	B1	2004-05-11	Kiraly , et al.	709/229
/AWJ/	13	6801912	B2	2004-10-05	Moskowitz , et al.	707/10
/AWJ/	14	6915105	B2	2005-07-05	Masuda	434/350
/AWJ/	15	6973446	B2	2005-12-06	Mamitsuka , et al.	706/50
/AWJ/	16	7017153	B2	2006-03-21	Gouriou , et al.	717/158
/AWJ/	17	7052277	B2	2006-05-30	Kellman	434/118
/AWJ/	18	7222127	B1	2007-05-22	Bem , et al.	707/102
/AWJ/	19	7240335	B2	2007-07-03	Angel , et al.	717/130

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	Application Number		16584736
	Filing Date		2019-09-26
	First Named Inventor	Jasmin Cosic	
	Art Unit		
	Examiner Name		
	Attorney Docket Number		

/AWJ/	20	7249349	B2	2007-07-24	Hundt , et al.	717/130
/AWJ/	21	7424705	B2	2008-09-09	Lewis , et al.	717/148
/AWJ/	22	7478371	B1	2009-01-13	Gove	717/128
/AWJ/	23	7484205	B2	2009-01-27	Venkatapathy	717/158
/AWJ/	24	7721218	B2	2010-05-18	Awe , et al.	715/762
/AWJ/	25	7765537	B2	2010-07-27	Havin , et al.	717/166
/AWJ/	26	7797259	B2	2010-09-14	Jiang , et al.	706/21
/AWJ/	27	7840060	B2	2010-11-23	Podilchuk	382/155
/AWJ/	28	7925984	B2	2011-04-12	Awe , et al.	715/762
/AWJ/	29	7987144	B1	2011-07-26	Drissi , et al.	706/20
/AWJ/	30	8005828	B2	2011-08-23	Buchner , et al.	707/728

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

/AWJ/	31	8019699	B2	2011-09-13	Baxter	706/12
/AWJ/	32	8078556	B2	2011-12-13	Adi , et al.	706/16
/AWJ/	33	8090669	B2	2012-01-03	Shahani , et al.	706/14
/AWJ/	34	8137112	B2	2012-03-20	Woolf , et al.	434/323
/AWJ/	35	8166463	B2	2012-04-24	Gill , et al.	717/130
/AWJ/	36	8195674	B1	2012-06-05	Bem , et al.	707/749
/AWJ/	37	8196119	B2	2012-06-05	Gill , et al.	717/130
/AWJ/	38	8244730	B2	2012-08-14	Gupta	707/737
/AWJ/	39	8251704	B2	2012-08-28	Woolf , et al.	434/322
/AWJ/	40	8266608	B2	2012-09-11	Hecht , et al.	717/158
/AWJ/	41	8364612	B2	2013-01-29	Van Gael , et al.	706/12

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

/AWJ/	42	8386401	B2	2013-02-26	Virkar , et al.	706/12
/AWJ/	43	8397227	B2	2013-03-12	Fan , et al.	717/166
/AWJ/	44	8464225	B2	2013-06-11	Greifeneder	717/130
/AWJ/	45	8549359	B2	2013-10-01	Zheng	714/38.11
/AWJ/	46	8589414	B2	2013-11-19	Waite , et al.	707/749
/AWJ/	47	8655260	B2	2014-02-18	Yueh , et al.	434/351
/AWJ/	48	8667472	B1	2014-03-04	Molinari	717/130

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS

Remove

Examiner Initial*	Cite No	Publication Number	Kind Code1	Publication Date	Name of Patentee or Applicant of cited Document	~~Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear~~ USPC/CPC
/AWJ/	1	20120167057	A1	2012-06-28	Schmich; Christopher P. ; et al.	717/130
/AWJ/	2	20120150773	A1	2012-06-14	DiCorpo; Phillip ; et al.	706/12

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

/AWJ/	3	20080254429	A1	2008-10-16	Woolf; Susan D. ; et al.	434/322
/AWJ/	4	20040117771	A1	2004-06-17	Venkatapathy, Ramanathan	717/130
/AWJ/	5	20130238533	A1	2013-09-12	VIRKAR; Hemant ; et al.	G06N 99/005
/AWJ/	6	20060190930	A1	2006-08-24	Hecht; Daniel M. ; et al.	717/128
/AWJ/	7	20070006159	A1	2007-01-04	Hecht; Daniel Michael ; et al.	717/124
/AWJ/	8	20080281764	A1	2008-11-13	Baxter; Jonathan	706/12
/AWJ/	9	20100138370	A1	2010-06-03	Wu; Wang ; et al.	706/12
/AWJ/	10	20110218672	A1	2011-09-08	Maisonnier; Bruno ; et al.	700/246

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number3	Country Code2 i	Kind Code4	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T5
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T5
/AWJ/	1	About Event Tracing, retrieved from <URL: http://msdn.microsoft.com/en-us/library/aa363668(d=default,l=en-us,v=vs.85).aspx> on Jan 12, 2014, 2 pages	
/AWJ/	2	Add dynamic Java code to your application, retrieved from <URL: http://www.javaworld.com/article/2071777/design-patterns/add-dynamic-java-code-to-your-application.html> on May 13, 2014, 6 pages	
/AWJ/	3	BCEL, retrieved from <URL: http://commons.apache.org> on May 13, 2014, 2 pages	
/AWJ/	4	Artificial intelligence, retrieved from <URL: http://wikipedia.com> on May 21, 2014, 28 pages	
/AWJ/	5	Artificial neural network, retrieved from <URL: http://wikipedia.com> on May 21, 2014, 12 pages	
/AWJ/	6	Branch (computer science), retrieved from <URL: http://wikipedia.com> on May 6, 2014, 2 pages	
/AWJ/	7	Branch trace, retrieved from <URL: http://wikipedia.com> on May 28, 2014, 2 pages	
/AWJ/	8	C Function Call Conventions and the Stack, retrieved from <URL: http://www.csee.umbc.edu/~chang/cs313.s02/stack.shtml> on Jan 11, 2014, 4 pages	
/AWJ/	9	Call stack, retrieved from <URL: http://wikipedia.com> on Jan 11, 2014, 5 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

/AWJ/	10	Program compilation and execution flow, retrieved from <URL: http://cs.stackexchange.com/questions/6187/program-compilation-and-execution-flow> on Jan 9, 2014, 2 pages	
/AWJ/	11	Control unit, retrieved from <URL: http://wikipedia.com> on May 25, 2014, 2 pages	
/AWJ/	12	Create dynamic applications with javax.tools, retrieved from <URL: http://www.ibm.com/developerworks/library/j-jcomp/> on May 13, 2014, 11 pages	
/AWJ/	13	Creating a Debugging and Profiling Agent with JVMTI, retrieved from <URL: http://www.oracle.com/technetwork/articles/java/jvmti-136367.html> on Jan 11, 2014, 18 pages	
/AWJ/	14	Decision tree learning, retrieved from <URL: http://wikipedia.com> on Jun 24, 2014, 5 pages	
/AWJ/	15	Dynamic Bytecode Instrumentation, retrieved from <URL: http://www.drdobbs.com/tools/dynamic-bytecode-instrumentation/184406433> on Apr 26, 2014, 6 pages	
/AWJ/	16	Dynamic programming language, retrieved from <URL: http://wikipedia.com> on May 7, 2014, 4 pages	
/AWJ/	17	Dynamic recompilation, retrieved from <URL: http://wikipedia.com> on May 19, 2014, 4 pages	
/AWJ/	18	eval(), retrieved from <URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/eval> on May 7, 2014, 13 pages	
/AWJ/	19	Function, retrieved from <URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Function> on May 7, 2014, 11 pages	
/AWJ/	20	FunctionEnter2 Function, retrieved from <URL: https://msdn.microsoft.com/en-us/library/aa964981(d=default,l=en-us,v=vs.110).aspx> on Mar 19, 2014, 2 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	Application Number		16584736
	Filing Date		2019-09-26
	First Named Inventor	Jasmin Cosic	
	Art Unit		
	Examiner Name		
	Attorney Docket Number		

/AWJ/	21	FunctionLeave2 Function, retrieved from <URL: https://msdn.microsoft.com/en-us/library/aa964942(d=default,l=en-us,v=vs.110).aspx> on Mar 19, 2014, 1 pages	
/AWJ/	22	Heuristic, retrieved from <URL: http://wikipedia.com> on May 22, 2014, 5 pages	
/AWJ/	23	How to: Configure Trace Switches, retrieved from <URL: https://msdn.microsoft.com/en-us/library/vstudio/t06xyy08(d=default,l=en-us,v=vs.100).aspx> on Mar 19, 2014, 3 pages	
/AWJ/	24	Indirect branch, retrieved from <URL: http://wikipedia.com> on May 5, 2014, 1 pages	
/AWJ/	25	Instruction cycle, retrieved from <URL: http://wikipedia.com> on Apr 30, 2014, 2 pages	
/AWJ/	26	Instruction set, retrieved from <URL: http://wikipedia.com> on Apr 29, 2014, 6 pages	
/AWJ/	27	Interrupt, retrieved from <URL: http://wikipedia.com> on May 15, 2014, 5 pages	
/AWJ/	28	Introduction to Instrumentation and Tracing, retrieved from <URL: http://msdn.microsoft.com/en-us/library/aa983649(d=default,l=en-us,v=vs.71).aspx> on Jan 10, 2014, 2 pages	
/AWJ/	29	Java bytecode, retrieved from <URL: http://wikipedia.com> on May 27, 2014, 4 pages	
/AWJ/	30	Package java.util.logging, retrieved from <URL: http://docs.oracle.com/javase/1.5.0/docs/api/java/util/logging/package-summary.html> on Mar 19, 2014, 2 pages	
/AWJ/	31	javac - Java programming language compiler, retrieved from <URL: http://docs.oracle.com/javase/7/docs/technotes/tools/windows/javac.html> on May 13, 2014, 12 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

/AWJ/	32	Interface JavaCompiler, retrieved from <URL: http://wikipedia.com> on May 13, 2014, 4 pages	
/AWJ/	33	Javassist, retrieved from <URL: http://wikipedia.com> on May 13, 2014, 2 pages	
/AWJ/	34	Just-in-time compilation, retrieved from <URL: http://wikipedia.com> on May 19, 2014, 5 pages	
/AWJ/	35	JVM Tool Interface, retrieved from <URL: http://docs.oracle.com/javase/7/docs/platform/jvmti/jvmti.html> on Apr 26, 2014, 127 pages	
/AWJ/	36	Machine code, retrieved from <URL: http://wikipedia.com> on Apr 30, 2014, 3 pages	
/AWJ/	37	Metaprogramming, retrieved from <URL: http://wikipedia.com> on May 8, 2014, 3 pages	
/AWJ/	38	Microsoft.VisualBasic.Logging Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/ms128122(d=default,l=en-us,v=vs.110).aspx> on Jan 13, 2014, 1 pages	
/AWJ/	39	Pin (computer program), retrieved from <URL: http://wikipedia.com> on May 6, 2014, 3 pages	
/AWJ/	40	Processor register, retrieved from <URL: http://wikipedia.com> on Apr 29, 2014, 4 pages	
/AWJ/	41	Profiling (computer programming), retrieved from <URL: http://wikipedia.com> on Jan 10, 2014, 4 pages	
/AWJ/	42	Profiling Overview, retrieved from <URL: http://msdn.microsoft.com/en-us/library/bb384493(d=default,l=en-us,v=vs.110).aspx> on Jan 12, 2014, 6 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number		16584736
	Filing Date		2019-09-26
	First Named Inventor	Jasmin Cosic	
	Art Unit		
	Examiner Name		
	Attorney Docket Number		

/AWJ/	43	Program counter, retrieved from <URL: http://wikipedia.com> on Apr 24, 2014, 2 pages	
/AWJ/	44	ptrace, retrieved from <URL: http://wikipedia.com> on Apr 24, 2014, 2 pages	
/AWJ/	45	Reflection (computer programming), retrieved from <URL: http://wikipedia.com> on May 7, 2014, 5 pages	
/AWJ/	46	Class Runtime, retrieved from <URL: http://docs.oracle.com/javase/7/docs/api/java/lang/Runtime.html> on Jan 10, 2014, 10 pages	
/AWJ/	47	Self-modifying code, retrieved from <URL: http://wikipedia.com> on Jan 12, 2014, 6 pages	
/AWJ/	48	Subjective logic, retrieved from <URL: http://wikipedia.com> on May 22, 2014, 5 pages	
/AWJ/	49	System.CodeDom.Compiler Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/z6b99ydt(d=default,l=en-us,v=vs.110).aspx> on May 13, 2014, 2 pages	
/AWJ/	50	System.Reflection.Emit Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/xd5fw18y(d=default,l=en-us,v=vs.110).aspx> on May 13, 2014, 2 pages	

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature	/Andrew W Johns/	Date Considered	01/28/2021

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

EAST Search History

EAST Search History (Prior Art)

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
L1	48	cosic-jasmin$.in.	US-PGPUB; USPAT	OR	ON	2021/01/29 06:24
L2	106,694	(image or picture or video) near10 (correlat$6 or associat$6) near10 (instruct$5 or control$5)	US-PGPUB; USPAT	OR	ON	2021/01/29 06:24
L3	13	L1 and L2	US-PGPUB; USPAT; USOCR	OR	ON	2021/01/29 06:24
L4	4,687,949	(instruction or control$4 or operat$4) near5 device	US-PGPUB; USPAT	OR	ON	2021/01/29 06:30
L5	609,036	(correlat$5 or associat$5) near5 (image or picture or video)	US-PGPUB; USPAT	OR	ON	2021/01/29 06:30
L6	9,984	(learn$4 or train$4) near8 L5	US-PGPUB; USPAT	OR	ON	2021/01/29 06:30
L7	472	L6 same L4	US-PGPUB; USPAT	OR	ON	2021/01/29 06:30
L8	6,800	(g06k9/00355 or g06k9/00389 or g06k9/66).cpc.	US-PGPUB; USPAT	OR	ON	2021/01/29 06:30
L9	40	L7 and L8	US-PGPUB; USPAT	OR	ON	2021/01/29 06:30
L10	7,872	g06t2207/20081.cpc.	US-PGPUB; USPAT	OR	ON	2021/01/29 06:35
L11	78	L7 and L10	US-PGPUB; USPAT	OR	ON	2021/01/29 06:35
L12	27,194	(g06f3/002 or g06f3/005 or g06f3/017).cpc.	US-PGPUB; USPAT	OR	ON	2021/01/29 06:40
L13	3	L7 and L12	US-PGPUB; USPAT	OR	ON	2021/01/29 06:40
L14	40,381	(g06n5/022 or g06n5/025 or g06n20/00).cpc.	US-PGPUB; USPAT	OR	ON	2021/01/29 06:41
L15	67	L7 and L14	US-PGPUB; USPAT	OR	ON	2021/01/29 06:41

1/29/2021 6:50:56 AM
C:\Users\ajohns\Documents\EAST\Workspaces\Applications\16\500\16584736.wsp

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS						Remove USPC/CPC
Examiner Initial*	Cite No	Patent Number	Kind Code1	Issue Date	Name of Patentee or Applicant of cited Document	~~Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear~~
/AWJ/	1	7113946	B2	2006-09-26	Cosic	707/8
/AWJ/	2	7117225	B2	2006-10-03	Cosic	707/102
/AWJ/	3	8335805	B2	2012-12-18	Cosic	707/805
/AWJ/	4	8417740	B2	2013-04-09	Cosic	707/805
/AWJ/	5	8572035	B2	2013-10-29	Cosic	707/625
/AWJ/	6	8655900	B2	2014-02-18	Cosic	707/760
/AWJ/	7	9047324	B2	2015-06-02	Cosic	G06F 17/30289
/AWJ/	8	9282309	B1	2016-03-08	Cosic	H04N 9/87

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

/AWJ/	9	9298749	B2	2016-03-29	Cosic	G06F 17/30289
/AWJ/	10	9367806	B1	2016-06-14	Cosic	G06N 5/022
/AWJ/	11	9443192	B1	2016-09-13	Cosic	G06N 3/088
/AWJ/	12	9582762	B1	2017-02-28	Cosic	G06N 5/04
/AWJ/	13	9595294	B2	2017-03-14	Cosic	G11B 27/036
/AWJ/	14	9697869	B2	2017-07-04	Cosic	G11B 27/036

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS Remove USPC/CPC

Examiner Initial*	Cite No	Publication Number	Kind Code[1]	Publication Date	Name of Patentee or Applicant of cited Document	~~Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear~~
/AWJ/	1	20030065662	A1	2003-04-03	Cosic	707/9
/AWJ/	2	20040194017	A1	2004-09-30	Cosic	707/501.1
/AWJ/	3	20050149517	A1	2005-07-07	Cosic	707/3

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

/AWJ/	4	20050149542	A1	2005-07-07	Cosic	707/100
/AWJ/	5	20050289105	A1	2005-12-29	Cosic	707/1
/AWJ/	6	20100023541	A1	2010-01-28	Cosic	707/101
/AWJ/	7	20100082536	A1	2010-04-01	Cosic	707/610
/AWJ/	8	20130218932	A1	2013-08-22	Cosic	G06F 17/30289
/AWJ/	9	20130226974	A1	2013-08-29	Cosic	G06F 17/30289
/AWJ/	10	20160140999	A1	2016-05-19	Cosic	G11B 27/036
/AWJ/	11	20160142650	A1	2016-05-19	Cosic	H04N 5/2624
/AWJ/	12	20160246819	A1	2016-08-25	Cosic	G06F 17/30289
/AWJ/	13	20160246850	A1	2016-08-25	Cosic	G06F 17/30528
/AWJ/	14	20160246868	A1	2016-08-25	Cosic	G06F 17/30595

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	Application Number		16584736
	Filing Date		2019-09-26
	First Named Inventor	Jasmin Cosic	
	Art Unit		
	Examiner Name		
	Attorney Docket Number		

/AWJ/	15	20160292185	A1	2016-10-06	Cosic	G06F 17/30289

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number3	Country Code2 i	Kind Code4	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T5
	~~1~~							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T5
	~~1~~		

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature	/Andrew W Johns/	Date Considered	01/28/2021

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

1 See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. 2 Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). 3 For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. 4 Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. 5 Applicant is to place a check mark here if English language translation is attached.

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

PTO/SB/08a (01-10)
Approved for use through 07/31/2012. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS Remove

Examiner Initial*	Cite No	Patent Number	Kind Code[1]	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1					

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS Remove USPC/CPC

Examiner Initial*	Cite No	Publication Number	Kind Code[1]	Publication Date	Name of Patentee or Applicant of cited Document	~~Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear~~
/AWJ/	1	20030026588	A1	2003-02-06	Elder, James H. ; et al.	386/46
/AWJ/	2	20040249774	A1	2004-12-09	Caid, William R. ; et al.	706/14
/AWJ/	3	20040267521	A1	2004-12-30	Cutler, Ross ; et al.	704/202
/AWJ/	4	20050245303	A1	2005-11-03	Graepel, Kurt Hartwig ; et al.	463/1
/AWJ/	5	20060047612	A1	2006-03-02	Stanley; Kenneth O. ; et al.	706/16

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

/AWJ/	6	20070058856	A1	2007-03-15	Boregowda; Lokesh R. ; et al.	382/159
/AWJ/	7	20070106633	A1	2007-05-10	Reiner; Bruce	707/1
/AWJ/	8	20080144893	A1	2008-06-19	GUO; Chun Biao ; et al.	382/118
/AWJ/	9	20080288259	A1	2008-11-20	Chambers; Robert L. ; et al.	704/275
/AWJ/	10	20090110061	A1	2009-04-30	LEE; Kang Yong ; et al.	375/240.02
/AWJ/	11	20090131152	A1	2009-05-21	Busse; Martin A.	463/23
/AWJ/	12	20090141969	A1	2009-06-04	Yu; Kai ; et al.	382/157
/AWJ/	13	20090222388	A1	2009-09-30	Hua; Wei ; et al.	706/12
/AWJ/	14	20090324010	A1	2009-12-31	Hou; Billy	382/103
/AWJ/	15	20100033780	A1	2010-02-11	Gitter; Kurt	359/9
/AWJ/	16	20100063949	A1	2010-03-11	EATON; JOHN ERIC ; et al.	706/14

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

/AWJ/	17	20100241595	A1	2010-09-23	Felsher; David Paul	705/400
/AWJ/	18	20100278420	A1	2010-11-04	Shet; Vinay Damodar ; et al.	382/156
/AWJ/	19	20110030031	A1	2011-02-03	Lussier; Paul ; et al.	726/1
/AWJ/	20	20130156345	A1	2013-06-20	Shmunk; Dmitry Valerievich	G06T 3/4053
/AWJ/	21	20130159021	A1	2013-06-20	Felsher; David Paul	G06F 19/322
/AWJ/	22	20130278631	A1	2013-10-24	Border; John N. ; et al.	G02B 27/017
/AWJ/	23	20140052717	A1	2014-02-20	FRANKS; Kasian ; et al.	G06F 17/30867
/AWJ/	24	20140161250	A1	2014-06-12	Leeds; Richard ; et al.	H04M 1/656
/AWJ/	25	20140177946	A1	2014-06-26	LIM; Kil-Taek ; et al.	G06K 9/00369
/AWJ/	26	20140207580	A1	2014-06-24	Minnis; William Charles ; et al.	G06Q 30/0269
/AWJ/	27	20150039304	A1	2015-02-05	Wein; Ron	G10L 25/78

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

/AWJ/	28	20150055821	A1	2015-02-26	Fotland; David Allen	G06K 9/3241

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number[3]	Country Code[2] i	Kind Code[4]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T[5]
	~~1~~							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T[5]
/AWJ/	1	ABOUT | OpenCV, retrieved from <URL: http://opencv.org/about.html> on Dec 13, 2014, 1 pages	
/AWJ/	2	Animetrics, Inc. | 3D Facial Recognition, retrieved from <URL: http://animetrics.com/> on Dec 13, 2014, 2 pages	
/AWJ/	3	Bag-of-words model, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
/AWJ/	4	Beier–Neely morphing algorithm, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 1 pages	
/AWJ/	5	Computer vision, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 7 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

/AWJ/	6		Convolutional neural network, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 5 pages	
/AWJ/	7		Decimation (signal processing), retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 3 pages	
/AWJ/	8		Digital image processing, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 3 pages	
/AWJ/	9		Dissolve (filmmaking), retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 2 pages	
/AWJ/	10		Facial recognition system, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 6 pages	
/AWJ/	11		Feature detection (computer vision), retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 3 pages	
/AWJ/	12		Feature extraction, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
/AWJ/	13		Gesture recognition, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 5 pages	
/AWJ/	14		GrabCut, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 1 pages	
/AWJ/	15		Image processing, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
/AWJ/	16		Image segmentation, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 12 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

/AWJ/	17	Image warping, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 2 pages	
/AWJ/	18	Inbetweening, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 2 pages	
/AWJ/	19	Interpolation, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 5 pages	
/AWJ/	20	Language model, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 4 pages	
/AWJ/	21	List of speech recognition software, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 4 pages	
/AWJ/	22	Livewire Segmentation Technique, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 2 pages	
/AWJ/	23	Morphing, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 2 pages	
/AWJ/	24	Motion estimation, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
/AWJ/	25	Muse: Face Morph || Mesh Warping, retrieved from <URL: http://alexwolfe.blogspot.com/2011/10/face-morph-mesh-warping.html> on Nov 19, 2015, 2 pages	
/AWJ/	26	n-gram, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 5 pages	
/AWJ/	27	Optical flow, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 4 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

/AWJ/	28	Outline of object recognition, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 7 pages	
/AWJ/	29	Phonotactics, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 3 pages	
/AWJ/	30	Recurrent neural network, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 5 pages	
/AWJ/	31	Sample rate conversion, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 2 pages	
/AWJ/	32	Simple interactive object extraction, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
/AWJ/	33	Speech recognition, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 12 pages	
/AWJ/	34	Speech segmentation, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 4 pages	
/AWJ/	35	Welcome to recognize-speech.com, retrieved from <URL: http://recognize-speech.com/> on Oct 18, 2015, 1 pages	
/AWJ/	36	Introduction Speech, retrieved from <URL: http://recognize-speech.com/speech> on Oct 18, 2015, 1 pages	
/AWJ/	37	Preprocessing, retrieved from <URL: http://recognize-speech.com/preprocessing> on Oct 18, 2015, 4 pages	
/AWJ/	38	Feature Extraction, retrieved from <URL: http://recognize-speech.com/feature-extraction> on Oct 18, 2015, 3 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

/AWJ/	39	Acoustic model, retrieved from <URL: http://recognize-speech.com/acoustic-model> on Oct 18, 2015, 2 pages	
/AWJ/	40	Video content analysis, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
/AWJ/	41	Video tracking, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 2 pages	
/AWJ/	42	Andrej Karpathy, Thomas Leung, George Toderici, Rahul Sukthankar, Sanketh Shetty, Li Fei-Fei, Large-scale Video Classification with Convolutional Neural Networks, Apr 14, 2014, 8 pages, Stanford University	
/AWJ/	43	Karen Simonyan, Andrew Zisserman, Two-Stream Convolutional Networks for Action Recognition in Videos, Nov 13, 2014, 11 pages, University of Oxford	

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature	/Andrew W Johns/	Date Considered	01/28/2021

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS Remove

Examiner Initial*	Cite No	Patent Number	Kind Code1	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	~~1~~					

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS Remove

Examiner Initial*	Cite No	Publication Number	Kind Code1	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	~~1~~					

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number3	Country Code2 i	Kind Code4	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T5
	~~1~~							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T5

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

/AWJ/	1	Tracing (software), retrieved from <URL: http://wikipedia.com> on Jan 10, 2014, 3 pages	
/AWJ/	2	Tree (data structure), retrieved from <URL: http://wikipedia.com> on Jun 24, 2014, 6 pages	
/AWJ/	3	PTRACE(2), retrieved from <URL: http://unixhelp.ed.ac.uk/CGI/man-cgi?ptrace> on Mar 19, 2014, 5 pages	
/AWJ/	4	Wevtutil, retrieved from <URL: http://technet.microsoft.com/en-us/library/cc732848(d=default,l=en-us,v=ws.11).aspx> on Apr 28, 2014, 5 pages	
/AWJ/	5	Intel Processor Trace, retrieved from <URL: https://software.intel.com/en-us/blogs/2013/09/18/processor-tracing> on Apr 28, 2014, 3 pages	
/AWJ/	6	YOUNGHOON JUNG, JAVA DYNAMICS Reflection and a lot more, Oct 10, 2012, 55 pages, Columbia University	
/AWJ/	7	AMITABH SRIVASTAVA, ALAN EUSTACE, ATOM A System for Building Customized Program Analysis Tools, May 3, 2004, 12 pages	
/AWJ/	8	MATHEW SMITHSON, KAPIL ANAND, APARNA KOTHA, KHALED ELWAZEER, NATHAN GILES, RAJEEV BARUA, Binary Rewriting without Relocation Information, Nov 10, 2010, 11 pages, University of Maryland	
/AWJ/	9	MAREK OLSZEWSKI, KEIR MIERTE, ADAM CZAJKOWSKI, ANGELA DEMLE BROWN, JIT Instrumentation - A Novel Approach To Dynamically Instrument Operating Systems, Feb 12, 2007, 14 pages, University of Toronto	

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature	/Andrew W Johns/	Date Considered	01/28/2021

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed
PTO/SB/08a (01-10)
Approved for use through 07/31/2012. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS Remove

Examiner Initial*	Cite No	Patent Number	Kind Code1	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	8996432		2015-03-31	Fu; Jicheng	
	2	6754631		2004-06-22	Din	
	3	9305216		2016-04-05	Mishra	
	4	5560011		1996-09-23	Uyama	
	5	6842877		2005-01-11	Robarts , et al.	
	6	7565340		2009-07-21	Herlocker , et al.	
	7	8261199		2012-09-04	Cradick , et al.	
	8	8266608		2012-09-11	Hecht , et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	9	9268454		2016-02-23	Hamilton, II , et al.	
	10	8595154		2013-11-26	Breckenridge , et al.	
	11	6408262		2002-06-18	Leerberg , et al.	
	12	9488984		2016-11-08	Williams , et al.	
	13	9158813		2015-10-13	Ntoulas , et al.	
	14	8356026		2013-01-15	Heimendinger	
	15	10162740		2018-12-25	Setty , et al.	

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS Remove

Examiner Initial*	Cite No	Publication Number	Kind Code1	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	20120290347		2012-11-15	ELAZOUNI; ASHRAF ; et al.	
	2	20150324685		2015-11-12	Bohn; Richard Esten ; et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	3	20100114746		2010-05-06	Bobbitt; Russell Patrick ; et al.	
	4	20120284026		2012-11-08	Cardillo; Peter S. ; et al.	
	5	20140211988		2014-07-31	Fan; Quanfu ; et al.	
	6	20150006171		2015-01-01	Westby; Michael C. ; et al.	
	7	20150264306		2015-09-17	Marilly; Emmanuel ; et al.	
	8	20110085734		2011-04-14	Berg; Jared S. ; et al.	
	9	20150269415		2015-09-24	Gelbman; Alexander	
	10	20160274187		2016-09-22	MENON; SANKARAN M. ; et al.	
	11	20150339213		2015-11-22	LEE; Christopher Stephen ; et al.	
	12	20150310041		2015-10-29	Kier; Scott ; et al.	
	13	20160328480		2016-11-10	Owens; Erich James ; et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	14	20140075249		2014-03-13	SATO; Shuhei ; et al.	
	15	20110270794		2011-11-03	Drory; Tal ; et al.	
	16	20070050719		2007-03-01	Lui; Philip ; et al.	
	17	20070061735		2007-03-15	Hoffberg; Steven M. ; et al.	
	18	20060184410		2006-08-17	Ramamurthy; Shankar ; et al.	
	19	20060265406		2006-11-23	Chkodrov; Gueorgui B. ; et al.	
	20	20090287643		2009-11-19	Corville; Allen O. ; et al.	
	21	20110007079		2011-01-13	Perez et al.	
	22	20070050606		2007-03-01	Ferren et al.	
	23	20090067727		2009-03-12	HIROHATA; Hitoshi	
	24	20130007532		2013-01-03	Miller et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	25	20050240412		2005-10-27	Fujita	
	26	20160167226		2016-06-16	Schnittman	
	27	20090136095		2009-05-28	Marcon et al.	
	28	20080215508		2008-09-04	Hanneman; Jeffrey E. ; et al.	
	29	20090044113		2009-02-12	Jones; Scott T. ; et al.	
	30	20130278501		2013-10-24	BULZACKI; Adrian	
	31	20190107839		2019-04-11	Parashar; Priyam ; et al.	
	32	20180120813		2018-05-03	COFFMAN; Valerie R. ; et al.	
	33	20180101146		2018-04-12	Hariharan; Rajaraman ; et al.	
	34	20160274187		2016-09-22	MENON; SANKARAN M. ; et al.	
	35	20140177946		2014-06-26	LIM; Kil-Taek ; et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	36	20160243701		2016-08-25	Gildert; Suzanne ; et al.	
	37	20120167057		2012-06-28	Schmich; Christopher P. ; et al.	
	38	20100278420		2010-11-04	Shet; Vinay Damodar ; et al.	
	39	20170161958		2017-06-08	EILAT; Eran	
	40	20160077166		2016-03-17	Morozov; Alexey ; et al.	
	41	20180068564		2018-03-08	TANIGAWA; TORU ; et al.	
	42	20120192096		2012-07-26	Bowman; Gordon Gregory ; et al.	
	43	20150269156		2015-09-24	Awadallah; Ahmed Hassan ; et al.	
	44	20160063118		2016-03-03	Campbell; Craig S. ; et al.	
	45	20140278754		2014-09-18	Cronin; Beau David ; et al.	
	46	20130329946		2013-12-12	ARCHIBALD; Fitzgerald John ; et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number3	Country Code2 i	Kind Code4	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T5
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T5
	1	Chen et al. "Case-Based Reasoning System and Artificial Neural Networks: A Review Neural Comput & Applic (2001) 10: pp 264-276, 13 pages	
	2	JOHN J. GREFENSTETTE, CONNIE LOGGIA RAMSEY, ALAN C. SCHULTZ, Learning Sequential Decision Rules Using Simulation Models and Competition, 1990, Navy Center for Applied Research in Artificial Intelligence, Naval Research Laboratory, Washington, DC, 27 pages	
	3	ALAN C. SCHULTZ, JOHN J. GREFENSTETTE, Using a Genetic Algorithm to Learn Behaviors for Autonomous Vehicles, 1992, Navy Center for Applied Research in Artificial Intelligence, Naval Research Laboratory, Washington, DC, 12 pages	
	4	Koppula et al., "Anticipating human activities using object affordances for reactive robotic response", IEEE TRAMI 2016, published 5 May 2015, 16 pages	
	5	Orme, "System design tips for entry level smartphones - part 3", found online at "https://community.arm.com/processors/b/blog/posts/system-design-tips-for-entry-level-smartphones---part-3", Oct 21, 2013, 12 pages	
	6	NOWOSTAWSKI, MARIUSZ et al., "Dynamic Demes Parallel Genetic Algorithm", May 13, 1999, 6 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	7	Stack Overflow, How can I quantify difference between two images?, accessed 2 August 2017 at https://stackoverflow.com/questions/189943/how-can-i-quantify-difference-between-two-images, 8 pages	
	8	SKOLICKI, ZBIGNIEW et al., "The Influence of Migration Sizes and Intervals on Island Models", June 29, 2005, 8 pages	
	9	Vahid Lari, et al., "Decentralized dynamic resource management support for massively parallel processor arrays", September 11, 2011, 8 pages	
	10	Vahid Lari, et al., "Distributed resource reservation in massively parallel processor arrays", May 16, 2011, 4 pages	
	11	Mohsen Hayati and Yazdan Shirvany, "Artificial Neural Network Approach for Short Term Load Forecasting for Illam Region", January 2007, 5 pages	
	12	Hasim Sak, Andrew Senior, and Francoise Beaufays, "Long Short-Term Memory Recurrent Neural Network Architectures for Large Scale Acoustic Modeling", January 2014, 5 pages	
	13	Jorg Walter and Klaus Schulten, "Implementation of self-organizing neural networks for visuo-motor control of an industrial robot", January 1993, 10 pages	
	14	Heikki Hyotyniemi and Aarno Lehtola, "A Universal Relation Database Interface for Knowledge Based Systems", April 1991, 5 pages	
	15	Mrissa, Michael, et al. "An avatar architecture for the web of things." IEEE Internet Computing 19.2 (2015): 30-38., 9 pages	
	16	Luck, Michael, and Ruth Aylett. "Applying artificial intelligence to virtual reality: Intelligent virtual environments." Applied Artificial Intelligence 14.1 (2000): 3-32., 30 pages	
	17	Terdjimi, Mehdi, et al. "An avatar-based adaptation workflow for the web of things." Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), 2016 IEEE 25th International Conference on. IEEE, 2016., 6 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	18	Bogdanovych, Anton, et al. "Authentic interactive reenactment of cultural heritage with 3D virtual worlds and artificial intelligence." Applied Artificial Intelligence 24.6 (2010): 617-647., 32 pages	
	19	Hernandez, Marco E. Perez, and Stephan Reiff-Marganiec. "Autonomous and self controlling smart objects for the future internet." Future internet of things and cloud (FiCloud), 2015 3rd international conference on. IEEE, 2015., 8 pages	
	20	Medini et al., "Building a Web of Things with Avatars", Managing the Web of Things (2017), 30 pages	
	21	Zhang, Yiying, Lei Guo, and Nicolas D. Georganas. "AGILE: An architecture for agent-based collaborative and interactive virtual environments." Proc. Workshop on Application Virtual Reality Technologies for Future Telecommunication System, IEEE Globecom '2000 Conference. 2000. (Year: 2000), 9 pages	
	22	Eladhari, Mirjam P., "Semi-Autonomous Avatars in Virtual Game Worlds," Pre-conference to the ECREA 2010 - 3rd European Communication Conference, Avatars and Humans. Representing Users in Digital Games, Hamburg, Germany, October 2010, Available from: 2012-01-21 (Year: 2012), 8 pages	

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature		Date Considered	

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Jasmin Cosic/	Date (YYYY-MM-DD)	2021-04-27
Name/Print	Jasmin Cosic	Registration Number	

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Electronic Acknowledgement Receipt

EFS ID:	42572197
Application Number:	16584736
International Application Number:	
Confirmation Number:	3341
Title of Invention:	MACHINE LEARNING FOR COMPUTING ENABLED SYSTEMS AND/OR DEVICES
First Named Inventor/Applicant Name:	Jasmin Cosic
Customer Number:	116094
Filer:	Jasmin Cosic
Filer Authorized By:	
Attorney Docket Number:	
Receipt Date:	27-APR-2021
Filing Date:	26-SEP-2019
Time Stamp:	22:37:23
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	no

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1	Information Disclosure Statement (IDS) Form (SB08)	IDS_Part5.pdf	621637 3de2289cca18ba72e2cd19f164236061367293ad	no	11

Warnings:

Information:

2	Non Patent Literature	NPL_Part5.pdf	18908685 8ef8652a5a8bfce5af8a83b28665d1184d168588	no	271

Warnings:

The page size in the PDF is too large. The pages should be 8.5 x 11 or A4. If this PDF is submitted, the pages will be resized upon entry into the Image File Wrapper and may affect subsequent processing

Information:

Total Files Size (in bytes):	19530322

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.
National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.
New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

Doc Code: DIST.E.FILE **Document Description: Electronic Terminal Disclaimer - Filed**	PTO/SB/26 U.S. Patent and Trademark Office Department of Commerce

Electronic Petition Request	**TERMINAL DISCLAIMER TO OBVIATE A DOUBLE PATENTING REJECTION OVER A "PRIOR" PATENT**
Application Number	16584736
Filing Date	26-Sep-2019
First Named Inventor	Jasmin Cosic
Attorney Docket Number	
Title of Invention	MACHINE LEARNING FOR COMPUTING ENABLED SYSTEMS AND/OR DEVICES

☒ Filing of terminal disclaimer does not obviate requirement for response under 37 CFR 1.111 to outstanding Office Action

☒ This electronic Terminal Disclaimer is not being used for a Joint Research Agreement.

Owner	Percent Interest
Jasmin Cosic	100%

The owner(s) with percent interest listed above in the instant application hereby disclaims, except as provided below, the terminal part of the statutory term of any patent granted on the instant application which would extend beyond the expiration date of the full statutory term of prior patent number(s)

10474934

as the term of said prior patent is presently shortened by any terminal disclaimer. The owner hereby agrees that any patent so granted on the instant application shall be enforceable only for and during such period that it and the prior patent are commonly owned. This agreement runs with any patent granted on the instant application and is binding upon the grantee, its successors or assigns.

In making the above disclaimer, the owner does not disclaim the terminal part of the term of any patent granted on the instant application that would extend to the expiration date of the full statutory term of the prior patent, "as the term of said prior patent is presently shortened by any terminal disclaimer," in the event that said prior patent later:
- expires for failure to pay a maintenance fee;
- is held unenforceable;
- is found invalid by a court of competent jurisdiction;
- is statutorily disclaimed in whole or terminally disclaimed under 37 CFR 1.321;
- has all claims canceled by a reexamination certificate;
- is reissued; or
- is in any manner terminated prior to the expiration of its full statutory term as presently shortened by any terminal disclaimer.

◉ Terminal disclaimer fee under 37 CFR 1.20(d) is included with Electronic Terminal Disclaimer request.

○ I certify, in accordance with 37 CFR 1.4(d)(4), that the terminal disclaimer fee under 37 CFR 1.20(d) required for this terminal disclaimer has already been paid in the above-identified application.

Applicant claims the following fee status:

◉ Small Entity

○ Micro Entity

○ Regular Undiscounted

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

THIS PORTION MUST BE COMPLETED BY THE SIGNATORY OR SIGNATORIES

I certify, in accordance with 37 CFR 1.4(d)(4) that I am:

○ An attorney or agent registered to practice before the Patent and Trademark Office who is of record in this application

Registration Number ____________

◉ A sole inventor

○ A joint inventor; I certify that I am authorized to sign this submission on behalf of all of the inventors as evidenced by the power of attorney in the application

○ A joint inventor; all of whom are signing this request

Signature	/Jasmin Cosic/
Name	Jasmin Cosic

*Statement under 37 CFR 3.73(b) is required if terminal disclaimer is signed by the assignee (owner).
Form PTO/SB/96 may be used for making this certification. See MPEP § 324.

Electronic Patent Application Fee Transmittal

Application Number:	16584736
Filing Date:	26-Sep-2019
Title of Invention:	MACHINE LEARNING FOR COMPUTING ENABLED SYSTEMS AND/OR DEVICES
First Named Inventor/Applicant Name:	Jasmin Cosic
Filer:	Jasmin Cosic
Attorney Docket Number:	

Filed as Small Entity

Filing Fees for **Utility under 35 USC 111(a)**

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Basic Filing:				
STATUTORY OR TERMINAL DISCLAIMER	2814	1	170	170
Pages:				
Claims:				
Miscellaneous-Filing:				
Petition:				
Patent-Appeals-and-Interference:				
Post-Allowance-and-Post-Issuance:				

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Extension-of-Time:				
Miscellaneous:				
	Total in USD ($)			**170**

Doc Code: DISQ.E.FILE
Document Description: Electronic Terminal Disclaimer – Approved

Application No.: 16584736

Filing Date: 26-Sep-2019

Applicant/Patent under Reexamination: Cosic

Electronic Terminal Disclaimer filed on April 28, 2021

☒ APPROVED

This patent is subject to a terminal disclaimer

☐ DISAPPROVED

Approved/Disapproved by: Electronic Terminal Disclaimer automatically approved by EFS-Web

U.S. Patent and Trademark Office

Electronic Acknowledgement Receipt

EFS ID:	42583886
Application Number:	16584736
International Application Number:	
Confirmation Number:	3341
Title of Invention:	MACHINE LEARNING FOR COMPUTING ENABLED SYSTEMS AND/OR DEVICES
First Named Inventor/Applicant Name:	Jasmin Cosic
Customer Number:	116094
Filer:	Jasmin Cosic
Filer Authorized By:	
Attorney Docket Number:	
Receipt Date:	28-APR-2021
Filing Date:	26-SEP-2019
Time Stamp:	19:14:50
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	yes
Payment Type	CARD
Payment was successfully received in RAM	$170
RAM confirmation Number	E20214RJ14441333
Deposit Account	
Authorized User	

The Director of the USPTO is hereby authorized to charge indicated fees and credit any overpayment as follows:

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1	Terminal Disclaimer-Filed (Electronic)	eTerminal-Disclaimer.pdf	33189 882053cd9c46e07552902f739c40f93453ee0e4a	no	2

Warnings:

Information:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
2	Fee Worksheet (SB06)	fee-info.pdf	30153 dc41128f5607a9d1973eb0a2c0c5b91973e3b47a	no	2

Warnings:

Information:

Total Files Size (in bytes):	63342

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.
National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.
New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
16/584,736	09/26/2019	Jasmin Cosic		3341

116094 7590 04/29/2021
Jasmin Cosic
108 Woodbury Street
Pawtucket, RI 02861

EXAMINER	
JOHNS, ANDREW W	

ART UNIT	PAPER NUMBER
2665	

MAIL DATE	DELIVERY MODE
04/29/2021	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Applicant-Initiated Interview Summary	**Application No.** 16/584,736	**Applicant(s)** Cosic, Jasmin		
	Examiner ANDREW W JOHNS	**Art Unit** 2665	**AIA (First Inventor to File) Status** Yes	**Page** **1 of 1**

All Participants (applicant, applicants representative, PTO personnel)	**Title**	**Type**
ANDREW W JOHNS	Primary Examiner	Telephonic
Jasmin Cosic	Inventor	

Date of Interview: 27 April 2021

Issues Discussed:

35 U.S.C. 101

Applicant proposed two alternative claims to address statutory double patenting rejection. Examiner indicated that both would overcome the outstanding double patenting rejection. The first proposed claim (21a in the attached proposal) would require further consideration to determine whether it raised a non-statutory double patenting issue and would require further consideration with respect to the prior art, as it includes claim scope not previously considered. The second proposed claims (21b in the attached proposal) would be subject to a non-statutory double patenting rejection, which could be overcome by filing a Terminal Disclaimer. Applicant asked if the second proposal (21b) could be entered by examiner's amendment if a fully copy of the amended claims was provided. Examiner agreed to enter the proposed claims by examiner's amendment on the condition that an acceptable terminal disclaimer was timely filed and the amended claims did not raise any issues under 35 USC 112.

☑ Attachment

/Andrew W Johns/ Primary Examiner, Art Unit 2665	

Applicant is reminded that a complete written statement as to the substance of the interview must be made of record in the application file. It is the applicants responsibility to provide the written statement, unless the interview was initiated by the Examiner and the Examiner has indicated that a written summary will be provided. See MPEP 713.04
Please further see:
MPEP 713.04
Title 37 Code of Federal Regulations (CFR) § 1.133 Interviews, paragraph (b)
37 CFR § 1.2 Business to be transacted in writing

Applicant recordation instructions: The formal written reply to the last Office action must include the substance of the interview. (See MPEP section 713.04). If a reply to the last Office action has already been filed, applicant is given a non-extendable period of the longer of one month or thirty days from this interview date, or the mailing date of this interview summary form, whichever is later, to file a statement of the substance of the interview.

Examiner recordation instructions: Examiners must summarize the substance of any interview of record. A complete and proper recordation of the substance of an interview should include the items listed in MPEP 713.04 for complete and proper recordation including the identification of the general thrust of each argument or issue discussed, a general indication of any other pertinent matters discussed regarding patentability and the general results or outcome of the interview, to include an indication as to whether or not agreement was reached on the issues raised.

Inventor: Jasmin Cosic
Serial No.: 16/584,736
Filing Date: September 26, 2019

Agenda for Interview on April 27, 2021

21a. A system comprising:

one or more processors; and

one or more non-transitory machine readable media storing machine readable code that, when executed by the one or more processors, causes the one or more processors to perform at least:

accessing a first one or more digital pictures correlated with a first one or more instruction sets for operating a first device, wherein the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device are learned in a first learning process.

21b. A system comprising:

one or more processors; and

one or more non-transitory machine readable media storing machine readable code that, when executed by the one or more processors, causes the one or more processors to perform at least:

receiving a first one or more digital pictures;

receiving a first one or more instruction sets for operating a first device; and

learning the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device.

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Inventor:	Jasmin Cosic	Confirmation No.: 3341
Title:	"MACHINE LEARNING FOR COMPUTING ENABLED SYSTEMS AND/OR DEVICES"	
Serial No.:	16/584,736	Filed: September 26, 2019
Examiner:	JOHNS, ANDREW W	Group Art Unit: 2665

Via EFS-Web May 1, 2021

Mail Stop Amendment
COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, VA 22313-1450

AMENDMENT

Dear Commissioner:

In response to non-final office action dated February 1, 2021 ("Office Action") and interview with the Examiner on April 27, 2021, the applicant submits this amendment and respectfully requests reconsideration of the claims.

Amendments to the Claims begin on page 2.

Remarks begin on page 17.

Listing of Claims

1 - 40 (canceled)

41. (new) A system comprising:

one or more processors; and

one or more non-transitory machine readable media storing machine readable code that, when executed by the one or more processors, causes the one or more processors to perform at least:

receiving or generating a first one or more digital pictures;

receiving or generating a first one or more instruction sets for operating a first device; and

learning the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device.

42. (new) The system of claim 41, wherein the receiving the first one or more digital pictures includes receiving the first one or more digital pictures from a picture capturing apparatus, and wherein the receiving the first one or more instruction sets for operating the first device includes receiving the first one or more instruction sets for operating the first device from: an application for operating the first device, a system for operating the first device, one or more microcontrollers, another one or more processors, or one or more actuators, and wherein the learning the first one or more digital pictures correlated with the first

one or more instruction sets for operating the first device includes storing the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device into or onto at least one of: at least one non-transitory machine readable medium of the one or more non-transitory machine readable media, another one or more non-transitory machine readable media, one or more volatile memories, one or more non-volatile memories, one or more storage devices, or one or more storage systems.

43. (new) The system of claim 41, wherein the learning the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device includes determining that the first one or more instruction sets for operating the first device temporally correspond to the first one or more digital pictures.

44. (new) The system of claim 41, wherein the machine readable code, when executed by the one or more processors, causes the one or more processors to further perform at least:

receiving or generating a new one or more digital pictures;

determining the first one or more instruction sets for operating the first device based on at least partial match between the new one or more digital pictures and the first one or more digital pictures; and

at least in response to the determining, causing the first device or a second device to perform one or more operations defined by the first one or more instruction sets for operating the first device.

45. (new) The system of claim 44, wherein the causing the first device or the second device to perform the one or more operations defined by the first one or more instruction sets for operating the first device includes executing the first one or more instruction sets for operating the first device.

46. (new) The system of claim 44, wherein the determining the first one or more instruction sets for operating the first device based on the at least partial match between the new one or more digital pictures and the first one or more digital pictures includes determining at least partial match between: one or more portions of the new one or more digital pictures that represent one or more recognized objects, and one or more portions of the first one or more digital pictures that represent one or more recognized objects.

47. (new) The system of claim 44, wherein the determining the first one or more instruction sets for operating the first device based on the at least partial match between the new one or more digital pictures and the first one or more digital pictures includes determining at least partial match between: one or more objects recognized in the new one or more digital pictures, and one or more objects recognized in the first one or more digital pictures.

48. (new) The system of claim 44, wherein the first one or more instruction sets for operating the first device include one or more information about one or more states of: the first device, or a portion of the first device.

49. (new) The system of claim 44, wherein the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device are included in a knowledgebase, and wherein the knowledgebase further includes a second one or more digital pictures correlated with a second one or more instruction sets for operating the first device, and wherein at least a portion of the first one or more digital pictures or at least a portion of the first one or more instruction sets for operating the first device is learned in a first learning process that includes operating the first device at least partially by a first user, and wherein at least a portion of the second one or more digital pictures or at least a portion of the second one or more instruction sets for operating the first device is learned in a second learning process that includes operating the first device at least partially by the first user, and wherein the first user is: a human user, or a non-human user.

50. (new) The system of claim 44, wherein the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device are included in a knowledgebase, and wherein the knowledgebase further includes a second one or more digital pictures correlated with a second one or more

more instruction sets for operating the first device, and wherein at least a portion of the first one or more digital pictures or at least a portion of the first one or more instruction sets for operating the first device is learned in a first learning process that includes operating the first device at least partially by a first user, and wherein at least a portion of the second one or more digital pictures or at least a portion of the second one or more instruction sets for operating the first device is learned in a second learning process that includes operating the first device at least partially by a second user, and wherein the first user is: a human user, or a non-human user, and wherein the second user is: a human user, or a non-human user.

51. (new) The system of claim 44, wherein the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device are included in a knowledgebase, and wherein the knowledgebase further includes a second one or more digital pictures correlated with a second one or more instruction sets for operating a third device, and wherein at least a portion of the first one or more digital pictures or at least a portion of the first one or more instruction sets for operating the first device is learned in a first learning process that includes operating the first device at least partially by a first user, and wherein at least a portion of the second one or more digital pictures or at least a portion of the second one or more instruction sets for operating the third device is learned in a second learning process that includes operating the third device at least partially by: the first user, or a second user, and wherein the first user is: a

human user, or a non-human user, and wherein the second user is: a human user, or a non-human user.

52. (new) The system of claim 44, wherein the new one or more digital pictures depict at least a portion of a surrounding of the first device, and wherein the first one or more instruction sets for operating the first device are applied to the first device, and wherein the first device is caused to perform the one or more operations defined by the first one or more instruction sets for operating the first device.

53. (new) The system of claim 44, wherein the new one or more digital pictures depict at least a portion of a surrounding of the second device, and wherein the first one or more instruction sets for operating the first device are applied to the second device, and wherein the second device is caused to perform the one or more operations defined by the first one or more instruction sets for operating the first device.

54. (new) The system of claim 44, wherein the new one or more digital pictures depict at least a portion of a surrounding of the second device, and wherein the first one or more instruction sets for operating the first device or a copy of the first one or more instruction sets for operating the first device are modified and applied to the second device, and wherein the second device is caused to perform one or more operations defined by: the modified the first one or more

instruction sets for operating the first device, or the modified the copy of the first one or more instruction sets for operating the first device.

55. (new) The system of claim 44, wherein the machine readable code, when executed by the one or more processors, causes the one or more processors to further perform at least:

modifying:

the first one or more instruction sets for operating the first device, or

a copy of the first one or more instruction sets for operating the first device, and wherein the determining the first one or more instruction sets for operating the first device based on the at least partial match between the new one or more digital pictures and the first one or more digital pictures includes:

determining the modified the first one or more instruction sets for operating the first device based on the at least partial match between the new one or more digital pictures and the first one or more digital pictures, or

determining the modified the copy of the first one or more instruction sets for operating the first device based on the at least partial match between the new one or more digital pictures and the first one or more digital pictures, and wherein the causing the first device or the second device to perform the one or more operations defined by the first one or more instruction sets for operating the first device includes:

causing the first device or the second device to perform one or more operations defined by the modified the first one or more instruction sets for operating the first device, or

causing the first device or the second device to perform one or more operations defined by the modified the copy of the first one or more instruction sets for operating the first device.

56. (new) The system of claim 44, wherein the machine readable code, when executed by the one or more processors, causes the one or more processors to further perform at least:

modifying at least one of: the first one or more digital pictures, a copy of the first one or more digital pictures, the new one or more digital pictures, or a copy of the new one or more digital pictures, and wherein the at least partial match between the new one or more digital pictures and the first one or more digital pictures is determined at least by: (i) determining at least partial match between the modified the new one or more digital pictures and the first one or more digital pictures, (ii) determining at least partial match between the modified the copy of the new one or more digital pictures and the first one or more digital pictures, (iii) determining at least partial match between the new one or more digital pictures and the modified the first one or more digital pictures, (iv) determining at least partial match between the new one or more digital pictures and the modified the copy of the first one or more digital pictures, (v) determining at least partial match between the modified the new one or more digital pictures

and the modified the first one or more digital pictures, (vi) determining at least partial match between the modified the copy of the new one or more digital pictures and the modified the copy of the first one or more digital pictures, (vii) determining at least partial match between the modified the new one or more digital pictures and the modified the copy of the first one or more digital pictures, or (viii) determining at least partial match between the modified the copy of the new one or more digital pictures and the modified the first one or more digital pictures.

57. (new) The system of claim 44, wherein the machine readable code, when executed by the one or more processors, causes the one or more processors to further perform at least:

modifying:

the first one or more digital pictures, or

a copy of the first one or more digital pictures, and wherein the learning the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device includes:

learning the modified the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device, or

learning the modified the copy of the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device, and wherein the determining the first one or more instruction sets for operating

the first device based on the at least partial match between the new one or more digital pictures and the first one or more digital pictures includes:

determining the first one or more instruction sets for operating the first device based on at least partial match between the new one or more digital pictures and the modified the first one or more digital pictures, or

determining the first one or more instruction sets for operating the first device based on at least partial match between the new one or more digital pictures and the modified the copy of the first one or more digital pictures.

58. (new) The system of claim 44, wherein the machine readable code, when executed by the one or more processors, causes the one or more processors to further perform at least:

modifying:

the first one or more instruction sets for operating the first device, or

a copy of the first one or more instruction sets for operating the first device, and wherein the learning the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device includes:

learning the first one or more digital pictures correlated with the modified the first one or more instruction sets for operating the first device, or

learning the first one or more digital pictures correlated with the modified the copy of the first one or more instruction sets for operating the first device, and wherein the determining the first one or more instruction sets for operating the

first device based on the at least partial match between the new one or more digital pictures and the first one or more digital pictures includes:

determining the modified the first one or more instruction sets for operating the first device based on the at least partial match between the new one or more digital pictures and the first one or more digital pictures, or

determining the modified the copy of the first one or more instruction sets for operating the first device based on the at least partial match between the new one or more digital pictures and the first one or more digital pictures, and wherein the causing the first device or the second device to perform the one or more operations defined by the first one or more instruction sets for operating the first device includes:

causing the first device or the second device to perform one or more operations defined by the modified the first one or more instruction sets for operating the first device, or

causing the first device or the second device to perform one or more operations defined by the modified the copy of the first one or more instruction sets for operating the first device.

59. (new) The system of claim 44, wherein the system further comprising:

an artificial intelligence system, and wherein the learning the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device includes learning, at least in part by the artificial intelligence system, the first one or more digital pictures correlated with the first

one or more instruction sets for operating the first device, and wherein the determining the first one or more instruction sets for operating the first device based on the at least partial match between the new one or more digital pictures and the first one or more digital pictures includes determining, at least in part by the artificial intelligence system, the first one or more instruction sets for operating the first device based on the at least partial match between the new one or more digital pictures and the first one or more digital pictures.

60. (new) The system of claim 44, wherein the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device include the first one or more digital pictures connected, using at least one or more connections, with the first one or more instruction sets for operating the first device.

61. (new) The system of claim 44, wherein the first one or more digital pictures are: one or more whole digital pictures, one or more representations of one or more whole digital pictures, one or more portions of at least one digital picture, one or more representations of one or more portions of at least one digital picture, one or more features, one or more representations of one or more features, one or more collections of pixels, or one or more collections of values, and wherein the new one or more digital pictures are: one or more whole digital pictures, one or more representations of one or more whole digital pictures, one or more portions of at least one digital picture, one or more representations of

one or more portions of at least one digital picture, one or more features, one or more representations of one or more features, one or more collections of pixels, or one or more collections of values.

62. (new) The system of claim 44, wherein the system further comprising:

a server that receives from the first device at least one of: the first one or more digital pictures, or the first one or more instruction sets for operating the first device, and wherein the second device receives from the server the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device, and wherein the second device is caused to perform the one or more operations defined by the first one or more instruction sets for operating the first device.

63. (new) The system of claim 44, wherein at least some elements of the system are included in: a single device, or multiple devices, and wherein the one or more processors include: one or more microcontrollers, one or more computing devices, or one or more electronic devices, and wherein the first device includes a robot, a vehicle, an appliance, an electronic device, or a mechanical machine, and wherein the second device includes a robot, a vehicle, an appliance, an electronic device, or a mechanical machine, and wherein the first one or more digital pictures include: one or more still digital pictures, or one or more motion digital pictures, and wherein the new one or more digital pictures include: one or more still digital pictures, or one or more motion digital pictures,

and wherein an instruction set of the first one or more instruction sets for operating the first device includes at least one of: only one instruction, a plurality of instructions, one or more inputs, one or more commands, one or more keywords, one or more symbols, one or more operators, one or more variables, one or more parameters, one or more characters, one or more numbers, one or more values, one or more signals, one or more binary bits, one or more functions, one or more function references, one or more objects, one or more object references, one or more data structures, one or more data structure references, one or more states, one or more representations of one or more states, one or more representations of one or more user inputs, one or more codes, one or more data, or one or more information.

64. (new) A method implemented using a computing system that includes one or more processors, the method comprising:

receiving or generating a first one or more digital pictures;

receiving or generating a first one or more instruction sets for operating a first device; and

learning the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device.

65. (new) A system comprising:

means for receiving or generating a first one or more digital pictures;

means for receiving or generating a first one or more instruction sets for operating a first device; and

means for learning the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device.

Remarks

Claims 1-40 are canceled. Claims 41-65 are new. Upon entry of this amendment, claims 41-65 will be pending in the present application. The applicant respectfully requests reconsideration and allowance of all claims.

I. Statutory Double Patenting

Claims 21-40 are rejected under 35 USC. § 101 as claiming the same invention as that of claims 1-20 of prior U.S. Patent No. 10,474,934. This is a statutory double patenting rejection. Claims 21-40 are word-for-word identical to corresponding claims 1-20 of the '934 patent, so that they are clearly claiming identical subject matter. [Office Action 5:17-20].

The applicant presents new claims 41-65 that do not recite identical subject matter as claims 1-20 of U.S. Patent No. 10,474,934. In the interview on April 27, 2021, the Examiner stated that the above-presented independent claim 41 overcomes the statutory double patenting rejection. The Examiner also advised filing a terminal disclaimer to U.S. Patent No. 10,474,934, which the applicant has done.

II. Interview Summary

The applicant thanks the Examiner for conducting a telephonic interview on April 27, 2021 with the applicant.

The Examiner agreed that the above-presented independent claim 41 is allowable subject to filing a terminal disclaimer to U.S. Patent No. 10,474,934.

III. Conclusion

The applicant respectfully submits that the amendments presented in this response should not be construed as an admission that the claims as previously presented are not patentable, or that the cited prior art references anticipate or render obvious the claims as previously presented. Indeed, the applicant respectfully submits that the claims as previously presented, and/or further amendments thereto, also contain patentable subject material. Thus, the applicant reserves the right to pursue the claims as previously presented, and/or further amendments thereto, in one or more continuation applications.

In view of the foregoing remarks, the applicant respectfully submits that the entire application is in condition for allowance. The applicant respectfully requests that a timely Notice of Allowance be issued in this case. If the Examiner would like to discuss any aspect of this application, the Examiner is respectfully requested to contact the undersigned at (317) 772-1312.

I hereby certify that this correspondence is being submitted electronically via EFS Web to the United States Patent and Trademark Office.

By /Jasmin Cosic/
Jasmin Cosic

Date submitted: May 1, 2021

Respectfully submitted,

/Jasmin Cosic/

Jasmin Cosic

Electronic Acknowledgement Receipt

EFS ID:	42612013
Application Number:	16584736
International Application Number:	
Confirmation Number:	3341
Title of Invention:	MACHINE LEARNING FOR COMPUTING ENABLED SYSTEMS AND/OR DEVICES
First Named Inventor/Applicant Name:	Jasmin Cosic
Customer Number:	116094
Filer:	Jasmin Cosic
Filer Authorized By:	
Attorney Docket Number:	
Receipt Date:	01-MAY-2021
Filing Date:	26-SEP-2019
Time Stamp:	19:17:37
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	no

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1		Answer.pdf	63421 7de0d0b0b010b0c8b281ec2df8337dec8215ad2c	yes	18

Multipart Description/PDF files in .zip description

Document Description	Start	End
Amendment/Req. Reconsideration-After Non-Final Reject	1	1
Claims	2	16
Applicant Arguments/Remarks Made in an Amendment	17	18

Warnings:

Information:

Total Files Size (in bytes):	63421

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.
National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.
New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

PTO/SB/06 (09-11)
Approved for use through 1/31/2014. OMB 0651-0032
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

PATENT APPLICATION FEE DETERMINATION RECORD Substitute for Form PTO-875	Application or Docket Number 16/584,736	Filing Date 09/26/2019	☐ To be Mailed

ENTITY: ☐ LARGE ☑ SMALL ☐ MICRO

APPLICATION AS FILED - PART I

FOR	(Column 1) NUMBER FILED	(Column 2) NUMBER EXTRA	RATE ($)	FEE ($)
☐ BASIC FEE (37 CFR 1.16(a), (b), or (c))	N/A	N/A	N/A	
☐ SEARCH FEE (37 CFR 1.16(k), (i), or (m))	N/A	N/A	N/A	
☐ EXAMINATION FEE (37 CFR 1.16(o), (p), or (q))	N/A	N/A	N/A	
TOTAL CLAIMS (37 CFR 1.16(i))	minus 20 =	*	x $50 =	
INDEPENDENT CLAIMS (37 CFR 1.16(h))	minus 3 =	*	x $230 =	
☐ APPLICATION SIZE FEE (37 CFR 1.16(s))	If the specification and drawings exceed 100 sheets of paper, the application size fee due is $310 ($155 for small entity) for each additional 50 sheets or fraction thereof. See 35 U.S.C. 41(a)(1)(G) and 37 CFR 1.16(s).			
☐ MULTIPLE DEPENDENT CLAIM PRESENT (37 CFR 1.16(j))				
* If the difference in column 1 is less than zero, enter "0" in column 2.			TOTAL	

APPLICATION AS AMENDED - PART II

AMENDMENT	05/01/2021	(Column 1) CLAIMS REMAINING AFTER AMENDMENT		(Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR	(Column 3) PRESENT EXTRA	RATE ($)	ADDITIONAL FEE ($)
	Total (37 CFR 1.16(i))	* 25	Minus	** 20	= 5	x $50 =	250
	Independent (37 CFR 1.16(h))	* 3	Minus	*** 3	= 0	x $240 =	0
	☐ Application Size Fee (37 CFR 1.16(s))						
	☐ FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))						
						TOTAL ADD'L FEE	250

AMENDMENT		(Column 1) CLAIMS REMAINING AFTER AMENDMENT		(Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR	(Column 3) PRESENT EXTRA	RATE ($)	ADDITIONAL FEE ($)
	Total (37 CFR 1.16(i))	*	Minus	**	=	x $0 =	
	Independent (37 CFR 1.16(h))	*	Minus	***	=	x $0 =	
	☐ Application Size Fee (37 CFR 1.16(s))						
	☐ FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))						
						TOTAL ADD'L FEE	

* If the entry in column 1 is less than the entry in column 2, write "0" in column 3.	LIE
** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, enter "20".	/MARQUITA D JONES/
*** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, enter "3".	
The "Highest Number Previously Paid For" (Total or Independent) is the highest number found in the appropriate box in column 1.	

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
16/584,736	09/26/2019	Jasmin Cosic		3341

116094 7590 05/05/2021

Jasmin Cosic
108 Woodbury Street
Pawtucket, RI 02861

EXAMINER	
JOHNS, ANDREW W	

ART UNIT	PAPER NUMBER
2665	

MAIL DATE	DELIVERY MODE
05/05/2021	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

NOTICE REQUIRING EXCESS CLAIMS FEES	**Application No.** 16/584,736	**Applicant(s)** Cosic, Jasmin
	Art Unit OED2	**AIA (FITF) Status** Yes

The excess claim(s) filed on 01 May 2021 is/are not accompanied by the appropriate payment of excess claims fees set forth in 37 CFR 1.16(h)-(j) or 1.492(d)-(f). Excess claims fees are required for each claim in independent form in excess of three (§ 1.16(h)), each claim (whether dependent or independent) in excess of twenty (note that § 1.75(c) indicates how multiple dependent claims are considered for fee calculation purposes) (§ 1.16(i)), and each application that contains a multiple dependent claim (§ 1.16(j)).

Since the application is not under a final rejection, applicant is given a time period of **TWO (2) MONTHS** from the mailing date of this notice to submit either: (1) the fee payment of $ _____, or (2) an amendment in compliance with 37 CFR 1.121 that cancels the excess claim(s), in order to avoid ABANDONMENT. Extensions of this time period may be granted under 37 CFR 1.136, unless the excess claim(s) was/were presented in a preliminary amendment.

☐ 1. The funds in Deposit Account No. _____ are insufficient to cover the entire fee due. The balance is due within the time period set forth in this notice. See note below regarding the appropriate service charge.

☐ 2. The Credit Card payment to cover the entire fee due to Account _____ (**Card type + last 4 digits ONLY**) was refused. The balance is due within the time period set forth in this notice. See note below regarding the appropriate service charge.

☐ 3. The amendment that includes the excess claim(s) has not been entered, since applicant has failed to remit (or authorize charge to a Deposit Account or Credit Card) the fee as indicated on the attached Patent Application Fee Determination Record (PTO/SB/06). Remittance or authorization is due within the time period set forth in this notice .

☐ 4. The fee submitted in this application is insufficient. A balance of $ _____ is due for presentation of excess claims (37 CFR 1.16(h)-(j) or 1.492(d)-(f)).

☑ 5. Other.

Explanation (*Provide specific details of the required correction in order to assist the applicant. Indicate whether a service charge has been added to the fee due*): ($250.00) Excess Claims Fees are due. Please see the 'Patent Application Fee Determination Record".

THE AMOUNT OF THE FEE(S) DUE IS SUBJECT TO CHANGE, GENERALLY ON OCTOBER 1 OF EACH YEAR (37 CFR 1.16, 1.21 & 1.492). THE **AMOUNT OF THE FEE(S) DUE IS DETERMINED AS OF THE DATE A COMPLETE REPLY WITH THE APPROPRIATE FEE(S) IS RECEIVED BY THE OFFICE** (37 CFR 1.8 & 1.10). BECAUSE THE AMOUNT DUE IS SUBJECT TO CHANGE, IT IS RECOMMENDED THAT APPLICANT CHECK THE CURRENT FEE SCHEDULE WHICH IS AVAILABLE ON THE USPTOS WEBSITE AT: ***http://www.uspto.gov/web/offices/ac/qs/ope/fees.htm***

Service Charges: There is a $50 service charge for processing each payment refused (including a check returned unpaid) or charged back by a financial institution (37 CFR 1.21(m)). There is a $25.00 service charge for each month when the balance of a deposit account is below $1000 at the end of the month (37 CFR 1.21(b)(2)).

Technical Support Staff (TSS): /MARQUITA D JONES/ Phone Number: (571)272-7263

Note to TSS: Please do NOT use this notice if the application is under a final rejection.

Electronic Patent Application Fee Transmittal

Application Number:	16584736
Filing Date:	26-Sep-2019
Title of Invention:	MACHINE LEARNING FOR COMPUTING ENABLED SYSTEMS AND/OR DEVICES
First Named Inventor/Applicant Name:	Jasmin Cosic
Filer:	Jasmin Cosic
Attorney Docket Number:	

Filed as Small Entity

Filing Fees for **Utility under 35 USC 111(a)**

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Basic Filing:				
Pages:				
Claims:				
CLAIMS IN EXCESS OF 20	2202	14	50	700
Miscellaneous-Filing:				
Petition:				
Patent-Appeals-and-Interference:				
Post-Allowance-and-Post-Issuance:				

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Extension-of-Time:				
Miscellaneous:				
	Total in USD ($)			**700**

Electronic Acknowledgement Receipt

EFS ID:	42762628
Application Number:	16584736
International Application Number:	
Confirmation Number:	3341
Title of Invention:	MACHINE LEARNING FOR COMPUTING ENABLED SYSTEMS AND/OR DEVICES
First Named Inventor/Applicant Name:	Jasmin Cosic
Customer Number:	116094
Filer:	Jasmin Cosic
Filer Authorized By:	
Attorney Docket Number:	
Receipt Date:	19-MAY-2021
Filing Date:	26-SEP-2019
Time Stamp:	12:33:56
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	yes
Payment Type	CARD
Payment was successfully received in RAM	$700
RAM confirmation Number	E20215IC35343888
Deposit Account	
Authorized User	

The Director of the USPTO is hereby authorized to charge indicated fees and credit any overpayment as follows:

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1	Fee Worksheet (SB06)	fee-info.pdf	30236 3929b08fdc90aa2c56e2b077290a88f16c9512c2	no	2

Warnings:

Information:

Total Files Size (in bytes):	30236

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

<u>New Applications Under 35 U.S.C. 111</u>
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.
<u>National Stage of an International Application under 35 U.S.C. 371</u>
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.
<u>New International Application Filed with the USPTO as a Receiving Office</u>
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

NOTICE OF ALLOWANCE AND FEE(S) DUE

116094 7590 05/25/2021

Jasmin Cosic
108 Woodbury Street
Pawtucket, RI 02861

EXAMINER	
JOHNS, ANDREW W	
ART UNIT	PAPER NUMBER
2665	

DATE MAILED: 05/25/2021

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
16/584,736	09/26/2019	Jasmin Cosic		3341

TITLE OF INVENTION: MACHINE LEARNING FOR COMPUTING ENABLED SYSTEMS AND/OR DEVICES

APPLN. TYPE	ENTITY STATUS	ISSUE FEE DUE	PUBLICATION FEE DUE	PREV. PAID ISSUE FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	SMALL	$600	$0.00	$0.00	$600	08/25/2021

THE APPLICATION IDENTIFIED ABOVE HAS BEEN EXAMINED AND IS ALLOWED FOR ISSUANCE AS A PATENT. PROSECUTION ON THE MERITS IS CLOSED. THIS NOTICE OF ALLOWANCE IS NOT A GRANT OF PATENT RIGHTS. THIS APPLICATION IS SUBJECT TO WITHDRAWAL FROM ISSUE AT THE INITIATIVE OF THE OFFICE OR UPON PETITION BY THE APPLICANT. SEE 37 CFR 1.313 AND MPEP 1308.

THE ISSUE FEE AND PUBLICATION FEE (IF REQUIRED) MUST BE PAID WITHIN THREE MONTHS FROM THE MAILING DATE OF THIS NOTICE OR THIS APPLICATION SHALL BE REGARDED AS ABANDONED. THIS STATUTORY PERIOD CANNOT BE EXTENDED. SEE 35 U.S.C. 151. THE ISSUE FEE DUE INDICATED ABOVE DOES NOT REFLECT A CREDIT FOR ANY PREVIOUSLY PAID ISSUE FEE IN THIS APPLICATION. IF AN ISSUE FEE HAS PREVIOUSLY BEEN PAID IN THIS APPLICATION (AS SHOWN ABOVE), THE RETURN OF PART B OF THIS FORM WILL BE CONSIDERED A REQUEST TO REAPPLY THE PREVIOUSLY PAID ISSUE FEE TOWARD THE ISSUE FEE NOW DUE.

HOW TO REPLY TO THIS NOTICE:

I. Review the ENTITY STATUS shown above. If the ENTITY STATUS is shown as SMALL or MICRO, verify whether entitlement to that entity status still applies.

If the ENTITY STATUS is the same as shown above, pay the TOTAL FEE(S) DUE shown above.

If the ENTITY STATUS is changed from that shown above, on PART B - FEE(S) TRANSMITTAL, complete section number 5 titled "Change in Entity Status (from status indicated above)".

For purposes of this notice, small entity fees are 1/2 the amount of undiscounted fees, and micro entity fees are 1/2 the amount of small entity fees.

II. PART B - FEE(S) TRANSMITTAL, or its equivalent, must be completed and returned to the United States Patent and Trademark Office (USPTO) with your ISSUE FEE and PUBLICATION FEE (if required). If you are charging the fee(s) to your deposit account, section "4b" of Part B - Fee(s) Transmittal should be completed and an extra copy of the form should be submitted. If an equivalent of Part B is filed, a request to reapply a previously paid issue fee must be clearly made, and delays in processing may occur due to the difficulty in recognizing the paper as an equivalent of Part B.

III. All communications regarding this application must give the application number. Please direct all communications prior to issuance to Mail Stop ISSUE FEE unless advised to the contrary.

IMPORTANT REMINDER: Maintenance fees are due in utility patents issuing on applications filed on or after Dec. 12, 1980. It is patentee's responsibility to ensure timely payment of maintenance fees when due. More information is available at www.uspto.gov/PatentMaintenanceFees.

PART B - FEE(S) TRANSMITTAL

Complete and send this form, together with applicable fee(s), by mail or fax, or via EFS-Web.

By mail, send to: Mail Stop ISSUE FEE
Commissioner for Patents
P.O. Box 1450
Alexandria, Virginia 22313-1450

By fax, send to: (571)-273-2885

INSTRUCTIONS: This form should be used for transmitting the ISSUE FEE and PUBLICATION FEE (if required). Blocks 1 through 5 should be completed where appropriate. All further correspondence including the Patent, advance orders and notification of maintenance fees will be mailed to the current correspondence address as indicated unless corrected below or directed otherwise in Block 1, by (a) specifying a new correspondence address; and/or (b) indicating a separate "FEE ADDRESS" for maintenance fee notifications.

CURRENT CORRESPONDENCE ADDRESS (Note: Use Block 1 for any change of address)

116094 7590 05/25/2021

Jasmin Cosic
108 Woodbury Street
Pawtucket, RI 02861

Note: A certificate of mailing can only be used for domestic mailings of the Fee(s) Transmittal. This certificate cannot be used for any other accompanying papers. Each additional paper, such as an assignment or formal drawing, must have its own certificate of mailing or transmission.

Certificate of Mailing or Transmission

I hereby certify that this Fee(s) Transmittal is being deposited with the United States Postal Service with sufficient postage for first class mail in an envelope addressed to the Mail Stop ISSUE FEE address above, or being transmitted to the USPTO via EFS-Web or by facsimile to (571) 273-2885, on the date below.

(Typed or printed name)

(Signature)

(Date)

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
16/584,736	09/26/2019	Jasmin Cosic		3341

TITLE OF INVENTION: MACHINE LEARNING FOR COMPUTING ENABLED SYSTEMS AND/OR DEVICES

APPLN. TYPE	ENTITY STATUS	ISSUE FEE DUE	PUBLICATION FEE DUE	PREV. PAID ISSUE FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	SMALL	$600	$0.00	$0.00	$600	08/25/2021

EXAMINER	ART UNIT	CLASS-SUBCLASS
JOHNS, ANDREW W	2665	382-157000

1. Change of correspondence address or indication of "Fee Address" (37 CFR 1.363).

❑ Change of correspondence address (or Change of Correspondence Address form PTO/SB/122) attached.

❑ "Fee Address" indication (or "Fee Address" Indication form PTO/SB/47; Rev 03-09 or more recent) attached. **Use of a Customer Number is required.**

2. For printing on the patent front page, list
(1) The names of up to 3 registered patent attorneys or agents OR, alternatively,
(2) The name of a single firm (having as a member a registered attorney or agent) and the names of up to 2 registered patent attorneys or agents. If no name is listed, no name will be printed.

1____________________
2____________________
3____________________

3. ASSIGNEE NAME AND RESIDENCE DATA TO BE PRINTED ON THE PATENT (print or type)

PLEASE NOTE: Unless an assignee is identified below, no assignee data will appear on the patent. If an assignee is identified below, the document must have been previously recorded, or filed for recordation, as set forth in 37 CFR 3.11 and 37 CFR 3.81(a). Completion of this form is NOT a substitute for filing an assignment.

(A) NAME OF ASSIGNEE

(B) RESIDENCE: (CITY and STATE OR COUNTRY)

Please check the appropriate assignee category or categories (will not be printed on the patent) : ❑ Individual ❑ Corporation or other private group entity ❑ Government

4a. Fees submitted: ❑Issue Fee ❑Publication Fee (if required) ❑Advance Order - # of Copies __________

4b. Method of Payment: *(Please first reapply any previously paid fee shown above)*

❑ Electronic Payment via EFS-Web ❑ Enclosed check ❑ Non-electronic payment by credit card (Attach form PTO-2038)

❑ The Director is hereby authorized to charge the required fee(s), any deficiency, or credit any overpayment to Deposit Account No. __________

5. **Change in Entity Status** (from status indicated above)

❑ Applicant certifying micro entity status. See 37 CFR 1.29

❑ Applicant asserting small entity status. See 37 CFR 1.27

❑ Applicant changing to regular undiscounted fee status.

NOTE: Absent a valid certification of Micro Entity Status (see forms PTO/SB/15A and 15B), issue fee payment in the micro entity amount will not be accepted at the risk of application abandonment.
NOTE: If the application was previously under micro entity status, checking this box will be taken to be a notification of loss of entitlement to micro entity status.
NOTE: Checking this box will be taken to be a notification of loss of entitlement to small or micro entity status, as applicable.

NOTE: This form must be signed in accordance with 37 CFR 1.31 and 1.33. See 37 CFR 1.4 for signature requirements and certifications.

Authorized Signature ____________________ Date ____________________

Typed or printed name ____________________ Registration No. ____________________

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
16/584,736	09/26/2019	Jasmin Cosic		3341

116094 7590 05/25/2021

Jasmin Cosic
108 Woodbury Street
Pawtucket, RI 02861

EXAMINER	
JOHNS, ANDREW W	
ART UNIT	**PAPER NUMBER**
2665	

DATE MAILED: 05/25/2021

Determination of Patent Term Adjustment under 35 U.S.C. 154 (b)

(Applications filed on or after May 29, 2000)

The Office has discontinued providing a Patent Term Adjustment (PTA) calculation with the Notice of Allowance.

Section 1(h)(2) of the AIA Technical Corrections Act amended 35 U.S.C. 154(b)(3)(B)(i) to eliminate the requirement that the Office provide a patent term adjustment determination with the notice of allowance. See Revisions to Patent Term Adjustment, 78 Fed. Reg. 19416, 19417 (Apr. 1, 2013). Therefore, the Office is no longer providing an initial patent term adjustment determination with the notice of allowance. The Office will continue to provide a patent term adjustment determination with the Issue Notification Letter that is mailed to applicant approximately three weeks prior to the issue date of the patent, and will include the patent term adjustment on the patent. Any request for reconsideration of the patent term adjustment determination (or reinstatement of patent term adjustment) should follow the process outlined in 37 CFR 1.705.

Any questions regarding the Patent Term Extension or Adjustment determination should be directed to the Office of Patent Legal Administration at (571)-272-7702. Questions relating to issue and publication fee payments should be directed to the Customer Service Center of the Office of Patent Publication at 1-(888)-786-0101 or (571)-272-4200.

OMB Clearance and PRA Burden Statement for PTOL-85 Part B

The Paperwork Reduction Act (PRA) of 1995 requires Federal agencies to obtain Office of Management and Budget approval before requesting most types of information from the public. When OMB approves an agency request to collect information from the public, OMB (i) provides a valid OMB Control Number and expiration date for the agency to display on the instrument that will be used to collect the information and (ii) requires the agency to inform the public about the OMB Control Number's legal significance in accordance with 5 CFR 1320.5(b).

The information collected by PTOL-85 Part B is required by 37 CFR 1.311. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 30 minutes to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, Virginia 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, Virginia 22313-1450. Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether disclosure of these records is required by the Freedom of Information Act.
2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.
3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.
4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).
5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.
6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).
7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.
8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspection or an issued patent.
9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Notice of Allowability	Application No. 16/584,736	Applicant(s) Cosic, Jasmin	
	Examiner ANDREW W JOHNS	Art Unit 2665	AIA (FITF) Status Yes

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address--

All claims being allowable, PROSECUTION ON THE MERITS IS (OR REMAINS) CLOSED in this application. If not included herewith (or previously mailed), a Notice of Allowance (PTOL-85) or other appropriate communication will be mailed in due course. **THIS NOTICE OF ALLOWABILITY IS NOT A GRANT OF PATENT RIGHTS.** This application is subject to withdrawal from issue at the initiative of the Office or upon petition by the applicant. See 37 CFR 1.313 and MPEP 1308.

1.☑ This communication is responsive to the amendment filed 01 May 2021 and the interview of 18 May 2021.
☐ A declaration(s)/affidavit(s) under **37 CFR 1.130(b)** was/were filed on ______.

2.☐ An election was made by the applicant in response to a restriction requirement set forth during the interview on ______; the restriction requirement and election have been incorporated into this action.

3.☑ The allowed claim(s) is/are 41-74 . As a result of the allowed claim(s), you may be eligible to benefit from the **Patent Prosecution Highway** program at a participating intellectual property office for the corresponding application. For more information, please see **http://www.uspto.gov/patents/init_events/pph/index.jsp** or send an inquiry to **PPHfeedback@uspto.gov.**

4.☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

Certified copies:

a) ☐All b) ☐ Some *c) ☐ None of the:

1. ☐ Certified copies of the priority documents have been received.
2. ☐ Certified copies of the priority documents have been received in Application No. ______ .
3. ☐ Copies of the certified copies of the priority documents have been received in this national stage application from the International Bureau (PCT Rule 17.2(a)).

* Certified copies not received: ______ .

Applicant has THREE MONTHS FROM THE "MAILING DATE" of this communication to file a reply complying with the requirements noted below. Failure to timely comply will result in ABANDONMENT of this application.
THIS THREE-MONTH PERIOD IS NOT EXTENDABLE.

5.☐ CORRECTED DRAWINGS (as "replacement sheets") must be submitted.
☐ including changes required by the attached Examiner's Amendment / Comment or in the Office action of Paper No./Mail Date ______ .

Identifying indicia such as the application number (see 37 CFR 1.84(c)) should be written on the drawings in the front (not the back) of each sheet. Replacement sheet(s) should be labeled as such in the header according to 37 CFR 1.121(d).

6.☐ DEPOSIT OF and/or INFORMATION about the deposit of BIOLOGICAL MATERIAL must be submitted. Note the attached Examiner's comment regarding REQUIREMENT FOR THE DEPOSIT OF BIOLOGICAL MATERIAL.

Attachment(s)

1.☑ Notice of References Cited (PTO-892)
2.☑ Information Disclosure Statements (PTO/SB/08), Paper No./Mail Date ______.
3.☐ Examiner's Comment Regarding Requirement for Deposit of Biological Material ______.
4.☑ Interview Summary (PTO-413), Paper No./Mail Date. ______.
5. ☑ Examiner's Amendment/Comment
6. ☑ Examiner's Statement of Reasons for Allowance
7. ☐ Other ______.

/Andrew W Johns/
Primary Examiner, Art Unit 2665

DETAILED ACTION

Notice of Pre-AIA or AIA Status

1. The present application, filed on or after March 16, 2013, is being examined under the first inventor to file provisions of the AIA.

Terminal Disclaimer

2. The terminal disclaimer filed on 28 April 2021 disclaiming the terminal portion of any patent granted on this application which would extend beyond the expiration date of Patent No. 10,474,934 has been reviewed and is accepted. The terminal disclaimer has been recorded.

EXAMINER'S AMENDMENT

3. An examiner's amendment to the record appears below. Should the changes and/or additions be unacceptable to applicant, an amendment may be filed as provided by 37 C.F.R. § 1.312. To ensure consideration of such an amendment, it MUST be submitted no later than the payment of the issue fee.

Authorization for this examiner's amendment was given in an interview with Jasmin Cosic on 18 May 2021.

The application has been amended as follows:

In the claims:

Cancel claims 21-40.

Amend claims 41, 46-47, 49-54, 56, 62, 64-65 as follows:

41. (currently amended) A system comprising:

one or more processors; and

one or more non-transitory machine readable media storing machine readable code that, when executed by the one or more processors, causes the one or more processors to perform at least:

receiving or generating a first one or more digital pictures, wherein the first one or more digital pictures depict at least a portion of a first device's surrounding;

receiving or generating a first one or more instruction sets for operating [[a]] the first device; and

learning the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device.

46. (currently amended) The system of claim 44, wherein the determining the first one or more instruction sets for operating the first device based on the at least partial match between the new one or more digital pictures and the first one or more digital pictures includes determining at least partial match between[[:]] one or more portions of the new one or more digital pictures that represent one or more ~~recognized~~ objects[[,]] and one or more portions of the first one or more digital pictures that represent one or more ~~recognized~~ objects.

47. (currently amended) The system of claim 44, wherein the determining the first one or more instruction sets for operating the first device based on the at least partial match

between the new one or more digital pictures and the first one or more digital pictures includes determining at least partial match between[[:]] one or more objects ~~recognized~~ detected in the new one or more digital pictures[[,]] and one or more objects ~~recognized~~ detected in the first one or more digital pictures.

49. (currently amended) The system of claim 44, wherein the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device are included in a knowledgebase, and wherein the knowledgebase further includes a second one or more digital pictures correlated with a second one or more instruction sets for operating the first device, and wherein at least a portion of the first one or more digital pictures or at least a portion of the first one or more instruction sets for operating the first device is learned in a first learning process that includes operating the first device at least partially by a first user, and wherein at least a portion of the second one or more digital pictures or at least a portion of the second one or more instruction sets for operating the first device is learned in a second learning process that includes operating the first device at least partially by the first user~~, and wherein the first user is: a human user, or a non-human user~~.

50. (currently amended) The system of claim 44, wherein the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device are included in a knowledgebase, and wherein the knowledgebase further includes a second one or more digital pictures correlated with a second one or more instruction sets for operating the first device, and wherein at least a portion of the first one or more digital pictures or at least a portion of the first one or more instruction sets for operating the first device is learned

in a first learning process that includes operating the first device at least partially by a first user, and wherein at least a portion of the second one or more digital pictures or at least a portion of the second one or more instruction sets for operating the first device is learned in a second learning process that includes operating the first device at least partially by a second user~~, and wherein the first user is: a human user, or a non-human user, and wherein the second user is: a human user, or a non-human user~~.

51. (currently amended) The system of claim 44, wherein the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device are included in a knowledgebase, and wherein the knowledgebase further includes a second one or more digital pictures correlated with a second one or more instruction sets for operating a third device, and wherein at least a portion of the first one or more digital pictures or at least a portion of the first one or more instruction sets for operating the first device is learned in a first learning process that includes operating the first device at least partially by a first user, and wherein at least a portion of the second one or more digital pictures or at least a portion of the second one or more instruction sets for operating the third device is learned in a second learning process that includes operating the third device at least partially by: the first user, or a second user~~, and wherein the first user is: a human user, or a non-human user, and wherein the second user is: a human user, or a non-human user~~.

52. (currently amended) The system of claim 44, wherein the new one or more digital pictures depict at least a portion of ~~a surrounding of~~ the first device's surrounding, and wherein the first one or more instruction sets for operating the first device are applied to the first

device, and wherein the first device is caused to perform the one or more operations defined by the first one or more instruction sets for operating the first device.

53. (currently amended) The system of claim 44, wherein the new one or more digital pictures depict at least a portion of ~~a surrounding of~~ the second device's surrounding, and wherein the first one or more instruction sets for operating the first device are applied to the second device, and wherein the second device is caused to perform the one or more operations defined by the first one or more instruction sets for operating the first device.

54. (currently amended) The system of claim 44, wherein the new one or more digital pictures depict at least a portion of ~~a surrounding of~~ the second device's surrounding, and wherein the first one or more instruction sets for operating the first device or a copy of the first one or more instruction sets for operating the first device are modified and applied to the second device, and wherein the second device is caused to perform one or more operations defined by: the modified the first one or more instruction sets for operating the first device, or the modified the copy of the first one or more instruction sets for operating the first device.

56. (currently amended) The system of claim 44, wherein the machine readable code, when executed by the one or more processors, causes the one or more processors to further perform at least:

modifying at least one of: the first one or more digital pictures, a copy of the first one or more digital pictures, the new one or more digital pictures, or a copy of the new one or more digital pictures, and wherein the at least partial match between the new one or more digital pictures and the first one or more digital pictures ~~is determined at least by~~ includes: (i)

~~determining~~ at least partial match between the modified the new one or more digital pictures and the first one or more digital pictures, (ii) ~~determining~~ at least partial match between the modified the copy of the new one or more digital pictures and the first one or more digital pictures, (iii) ~~determining~~ at least partial match between the new one or more digital pictures and the modified the first one or more digital pictures, (iv) ~~determining~~ at least partial match between the new one or more digital pictures and the modified the copy of the first one or more digital pictures, (v) ~~determining~~ at least partial match between the modified the new one or more digital pictures and the modified the first one or more digital pictures, (vi) ~~determining~~ at least partial match between the modified the copy of the new one or more digital pictures and the modified the copy of the first one or more digital pictures, (vii) ~~determining~~ at least partial match between the modified the new one or more digital pictures and the modified the copy of the first one or more digital pictures, or (viii) ~~determining~~ at least partial match between the modified the copy of the new one or more digital pictures and the modified the first one or more digital pictures.

62. (currently amended) The system of claim 44, wherein the system further comprising:

a server that receives from the first device at least one of: the first one or more digital pictures, or the first one or more instruction sets for operating the first device, and wherein the second device receives from the server at least one of: the first one or more digital pictures, or ~~correlated with~~ the first one or more instruction sets for operating the first device, and

wherein the second device is caused to perform the one or more operations defined by the first one or more instruction sets for operating the first device.

64. (currently amended) A method implemented using a computing system that includes one or more processors, the method comprising:

receiving or generating a first one or more digital pictures<u>, wherein the first one or more digital pictures depict at least a portion of a first device's surrounding</u>;

receiving or generating a first one or more instruction sets for operating [[a]] <u>the</u> first device; and

learning the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device.

65. (currently amended) A system comprising:

means for receiving or generating a first one or more digital pictures<u>, wherein the first one or more digital pictures depict at least a portion of a first device's surrounding</u>;

means for receiving or generating a first one or more instruction sets for operating [[a]] <u>the</u> first device; and

means for learning the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device.

Add new claims 66-74:

66. (new) The system of claim 44, wherein the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device are included

in a knowledgebase, and wherein the knowledgebase further includes a second one or more digital pictures correlated with a second one or more instruction sets for operating a third device, and wherein the machine readable code, when executed by the one or more processors, causes the one or more processors to further perform at least:

receiving or generating an additional one or more digital pictures;

determining the second one or more instruction sets for operating the third device based on at least partial match between the additional one or more digital pictures and the second one or more digital pictures; and

at least in response to the determining the second one or more instruction sets for operating the third device, causing a fourth device to perform one or more operations defined by the second one or more instruction sets for operating the third device.

67. (new) The system of claim 44, wherein the at least partial match between the new one or more digital pictures and the first one or more digital pictures is at least in part based on a product of one or more values that represent one or more portions of the new one or more digital pictures and one or more values that represent one or more portions of the first one or more digital pictures.

68. (new) The system of claim 44, wherein the generating the new one or more digital pictures includes:

detecting one or more objects in the first device's surrounding or the second device's surrounding;

generating one or more representations of the one or more objects; and

generating the new one or more digital pictures that include the one or more

representations of the one or more objects.

69. (new) The system of claim 68, wherein the detecting the one or more objects in the first device's surrounding or the second device's surrounding includes detecting at least one or more locations of the one or more objects, and wherein the new one or more digital pictures include a top-down view of the one or more representations of the one or more objects.

70. (new) The system of claim 44, wherein the generating the first one or more instruction sets for operating the first device includes generating the first one or more instruction sets for operating the first device as one or more representations of a second one or more instruction sets for operating the first device.

71. (new) The system of claim 59, wherein the artificial intelligence system includes: one or more inputs for receiving one or more digital pictures, and one or more outputs for providing one or more instruction sets, and wherein the learning, at least in part by the artificial intelligence system, the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device includes applying the first one or more digital pictures to the inputs of the artificial intelligence system, and wherein the determining, at least in part by the artificial intelligence system, the first one or more instruction sets for operating

the first device based on the at least partial match between the new one or more digital pictures and the first one or more digital pictures includes:

applying the new one or more digital pictures to the inputs of the artificial intelligence system; and

receiving the first one or more instruction sets for operating the first device from the outputs of the artificial intelligence system.

72. (new) The system of claim 71, wherein the one or more inputs for receiving the one or more digital pictures include: one input for one digital picture, multiple inputs for multiple digital pictures, one input for one portion of one digital picture, multiple inputs for multiple portions of one digital picture, multiple inputs for multiple portions of multiple digital pictures, an input for one representation of one digital picture, multiple inputs for multiple representations of multiple digital pictures, one input for one representation of one portion of one digital picture, multiple inputs for multiple representations of multiple portions of one

digital picture, or multiple inputs for multiple representations of multiple portions of multiple digital pictures.

73. (new) The system of claim 59, wherein the artificial intelligence system includes: a knowledgebase, or a neural network.

74. (new) The system of claim 44, wherein the first one or more digital pictures that depict the at least the portion of the first device's surrounding include one or more digital pictures that depict a representation of the at least the portion of the first device's surrounding.

Reasons for Allowance

4. The following is an examiner's statement of reasons for allowance: The terminal disclaimer, filed 28 April 2021, obviates any non-statutory double patenting issues with Pat. No. 10,474,934. Additionally, the examiner's amendment, set forth above, distinguishes the instant claims from the claims of Pat. No. 9,864,933 because the claims of the '933 patent do not stipulate that the first one or more pictures depict at least a portion of the first device's surrounding, as now required by the claims of the instant application.

In addition, the prior art fails to teach or fairly suggest the claimed invention. Specifically, the prior art does not teach or suggest receiving or generating a first one or more digital pictures that depict at least a portion of a first device's surrounding, receiving or generating a first one or more instructions sets for operating the first device and learning the first picture(s)correlated with the first instruction set(s), as claimed. While Gelbman (US 2015/0269415 A1, cited in the IDS filed 27 April 2021) determines an action to be initiated by an instrument corresponding to a captured image (Abstract), and includes a learning process

for unidentified items (344 in Figure 3B), the learning process of Gelbman (described in paragraph [0029]) only adds the unidentified object to the database by a manual or formal approval process. There is no suggestion in Gelbman that a correlation between the image and an instruction set for instrument is learned. Similarly, while Kaehler et al. (US 2019/0034765 A1) teach using training images to train a neural network with representations of virtual devices and pointers, as described at paragraph [0078], the learning associates a UI event with the image. As described, the UI event is an activation or non-activation of a virtual device. There is no suggestion in Kaehler et al. of learning a correlation between the image and any instruction set(s) for operating a device, as claimed.

Any comments considered necessary by applicant must be submitted no later than the payment of the issue fee and, to avoid processing delays, should preferably accompany the issue fee. Such submissions should be clearly labeled "Comments on Statement of Reasons for Allowance."

5. Claims 41-59, 60-63, 64-65, 66-70, 71-73, and 74 (renumbered for issue as 1-19, 23-26, 33-34, 27-31, 20-22, and 32, respectively) are allowed.

Conclusion

6. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Andrew Johns whose telephone number is (571) 272-7391. The examiner in normally available Monday through Friday, typically between 6:15 am and 2:45 pm Eastern Time. The examiner may also be contacted by e-mail using the address: andrew.johns@uspto.gov. (Applicant is reminded of the Office policy regarding e-mail communications. See M.P.E.P. § 502.03)

If attempts to reach the examiner are unsuccessful, the examiner's supervisor, Edward Urban, can be reached at (571) 272-7899. The fax phone number for this art unit is (571) 273-

8300. In order to ensure prompt delivery to the examiner, all unofficial communications should be clearly labeled as "Draft" or "Unofficial."

A. Johns
19 May 2021

/Andrew W Johns/
Primary Examiner, Art Unit 2665

Examiner-Initiated Interview Summary	**Application No.** 16/584,736	**Applicant(s)** Cosic, Jasmin		
	Examiner ANDREW W JOHNS	**Art Unit** 2665	**AIA (First Inventor to File) Status** Yes	**Page** 1 of 2

All Participants (applicant, applicants representative, PTO personnel)	**Title**	**Type**
ANDREW W JOHNS	Primary Examiner	Telephonic
Jasmin Cosic	Inventor	

Date of Interview: 18 May 2021

Issues Discussed:

Proposed Amendment(s)

Applicant proposed amending the independent claims of the instant application to stipulate that the one or more digital pictures depict at least a portion of the surroundings of the first device. Examiner agreed that this language was not anticipated or obvious in view of the Pat. No. 9,864,933. In the '933 patent the pictures are received from the renderer and represent a virtual space created by the application program , which cannot be construed as being the same or equivalent to the surroundings of the device itself. Therefore, this proposed language would clearly distinguish the instant claims from the claims of the '933 patent.

Applicant also proposed amendments to claims 46-47, 49-54, 56, and 62 to either correct minor issues therein or to better conform with the amended independent claims, and further requested the addition of new dependent claims 66-74. Examiner agreed to enter these additional changes and new claims by examiner's amendment to expedite allowance of the application. (All proposed amendments attached hereto.)

Non-statutory Double Patenting

Examiner noted that while the terminal disclaimer filed 28 April 2021 obviated any potential non-statutory double patenting issues with respect to Pat. No. 10,474,934, a new potential double patenting issue was discovered during the updated and expanded search. In particular, examiner noted that each of the limitations of claim 41 appeared to be anticipated by limitations in claim 1 of Pat. No. 9,864,933. Applicant reviewed claims and argued that the '933 patent was directed towards a substantially different invention. Specifically, applicant argued that the instruction set(s) in the instant application are for operating a physical device in a real world situation, whereas the instruction set(s) in the '933 patent are used to operate an avatar (i.e., a digital representation in a virtual environment) in an application program (i.e., software, not a physical device). The examiner acknowledged these arguments, but noted that the claims of the instant application broadly read on the specifics of the '933 patent. Specifically, the examiner noted that "operating the avatar of the application program" does not happen in a vacuum - there must be a device (such as a computer or virtual reality device) that is running the application program and producing the avatar, so that the instruction set(s) of the '933 patent fall within the broad scope of the requirement of the instant claim that they operate a first device. Examiner indicated that because applicant did not wish to file a TD with respect to the '933 patent, the instant claims would need to be amended to distinguish over the claims of the '933 patent. After some discussion of possible changes to the claims of the instant application, agreement was reached regarding language that would clearly distinguish the claims of the instant application from the claims of the '933 patent.

Examiner-Initiated Interview Summary	**Application No.** 16/584,736	**Applicant(s)** Cosic, Jasmin		
	Examiner ANDREW W JOHNS	**Art Unit** 2665	**AIA (First Inventor to File) Status** Yes	**Page** **2 of 2**

☑ Attachment

/Andrew W Johns/ Primary Examiner, Art Unit 2665	

Applicant is reminded that a complete written statement as to the substance of the interview must be made of record in the application file. It is the applicants responsibility to provide the written statement, unless the interview was initiated by the Examiner and the Examiner has indicated that a written summary will be provided. See MPEP 713.04
Please further see:
MPEP 713.04
Title 37 Code of Federal Regulations (CFR) § 1.133 Interviews, paragraph (b)
37 CFR § 1.2 Business to be transacted in writing

Applicant recordation instructions: It is not necessary for applicant to provide a separate record of the substance of interview.

Examiner recordation instructions: Examiners must summarize the substance of any interview of record. A complete and proper recordation of the substance of an interview should include the items listed in MPEP 713.04 for complete and proper recordation including the identification of the general thrust of each argument or issue discussed, a general indication of any other pertinent matters discussed regarding patentability and the general results or outcome of the interview, to include an indication as to whether or not agreement was reached on the issues raised.

Notice of References Cited	Application/Control No. 16/584,736	Applicant(s)/Patent Under Reexamination Cosic, Jasmin	
	Examiner ANDREW W JOHNS	Art Unit 2665	Page 1 of 1

U.S. PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Name	CPC Classification	US Classification
*	A	US-9,864,933-B1	01-2018	Cosic	G06K9/52	1/1
*	B	US-2019/0034765-A1	01-2019	Kaehler et al.	G02B27/0172	1/1
	C					
	D					
	E					
	F					
	G					
	H					
	I					
	J					
	K					
	L					
	M					

FOREIGN PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Country	Name	CPC Classification
	N					
	O					
	P					
	Q					
	R					
	S					
	T					

NON-PATENT DOCUMENTS

*		Include as applicable: Author, Title Date, Publisher, Edition or Volume, Pertinent Pages)
	U	
	V	
	W	
	X	

*A copy of this reference is not being furnished with this Office action. (See MPEP § 707.05(a).)
Dates in MM-YYYY format are publication dates. Classifications may be US or foreign.

Index of Claims	Application/Control No. 16/584,736	Applicant(s)/Patent Under Reexamination Cosic, Jasmin
	Examiner ANDREW W JOHNS	Art Unit 2665

✓	Rejected		-	Cancelled		N	Non-Elected		A	Appeal
=	Allowed		÷	Restricted		I	Interference		O	Objected

CLAIMS

☐ Claims renumbered in the same order as presented by applicant ☐ CPA ☑ T.D. ☐ R.1.47

CLAIM		DATE								
Final	Original	01/29/2021	05/19/2021							
-	21	✓	-							
-	22	✓	-							
-	23	✓	-							
-	24	✓	-							
-	25	✓	-							
-	26	✓	-							
-	27	✓	-							
-	28	✓	-							
-	29	✓	-							
-	30	✓	-							
-	31	✓	-							
-	32	✓	-							
-	33	✓	-							
-	34	✓	-							
-	35	✓	-							
-	36	✓	-							
-	37	✓	-							
-	38	✓	-							
-	39	✓	-							
-	40	✓	-							
1	41		=							
2	42		=							
3	43		=							
4	44		=							
5	45		=							
6	46		=							
7	47		=							
8	48		=							
9	49		=							
10	50		=							
11	51		=							
12	52		=							
13	53		=							
14	54		=							
15	55		=							
16	56		=							
17	57		=							
18	58		=							
19	59		=							
23	60		=							
24	61		=							
25	62		=							

Index of Claims	Application/Control No. 16/584,736	Applicant(s)/Patent Under Reexamination Cosic, Jasmin
	Examiner ANDREW W JOHNS	Art Unit 2665

CLAIM		DATE								
Final	Original	01/29/2021	05/19/2021							
26	63		=							
33	64		=							
34	65		=							
27	66		=							
28	67		=							
29	68		=							
30	69		=							
31	70		=							
20	71		=							
21	72		=							
22	73		=							
32	74		=							

Issue Classification	Application/Control No.	Applicant(s)/Patent Under Reexamination
	16/584,736	Cosic, Jasmin
	Examiner	**Art Unit**
	ANDREW W JOHNS	2665

CPC						
Symbol					**Type**	**Version**
G06K	/	9	/	66	F	2013-01-01
G06K	/	9	/	6202	I	2013-01-01
G06T	/	7	/	70	I	2017-01-01
G06F	/	9	/	30076	I	2013-01-01
G06N	/	20	/	00	I	2019-01-01
G06F	/	15	/	80	A	2013-01-01
G06F	/	15	/	78	A	2013-01-01
G06N	/	5	/	022	A	2013-01-01

CPC Combination Sets								
Symbol					**Type**	**Set**	**Ranking**	**Version**
	/		/					

NONE (Assistant Examiner) (Date)	**Total Claims Allowed:** 34	
/Andrew W Johns/ Primary Examiner, Art Unit 2665 19 May 2021 (Primary Examiner) (Date)	O.G. Print Claim(s) 1	O.G. Print Figure 31

U.S. Patent and Trademark Office

Part of Paper No.: 20210519

Issue Classification	Application/Control No. 16/584,736	Applicant(s)/Patent Under Reexamination Cosic, Jasmin
	Examiner ANDREW W JOHNS	Art Unit 2665

INTERNATIONAL CLASSIFICATION				
CLAIMED				
G06K9/66	/	9	/	66
G06N5/02	/	5	/	02

NON-CLAIMED				
	/		/	

US ORIGINAL CLASSIFICATION	
CLASS	SUBCLASS

CROSS REFERENCES(S)						
CLASS	SUBCLASS (ONE SUBCLASS PER BLOCK)					

NONE (Assistant Examiner) (Date)		Total Claims Allowed: 34	
/Andrew W Johns/ Primary Examiner, Art Unit 2665 (Primary Examiner)	19 May 2021 (Date)	O.G. Print Claim(s) 1	O.G. Print Figure 31

U.S. Patent and Trademark Office Part of Paper No.: 20210519

Issue Classification	Application/Control No.	Applicant(s)/Patent Under Reexamination
	16/584,736	Cosic, Jasmin
	Examiner	**Art Unit**
	ANDREW W JOHNS	2665

☐ Claims renumbered in the same order as presented by applicant ☐ CPA ☑ T.D. ☐ R.1.47

CLAIMS

Final	Original	Final	Original	Final	Original	Final	Original	Final	Original	Final	Original	Final	Original	Final	Original
-	21	-	30	-	39	8	48	17	57	27	66				
-	22	-	31	-	40	9	49	18	58	28	67				
-	23	-	32	1	41	10	50	19	59	29	68				
-	24	-	33	2	42	11	51	23	60	30	69				
-	25	-	34	3	43	12	52	24	61	31	70				
-	26	-	35	4	44	13	53	25	62	20	71				
-	27	-	36	5	45	14	54	26	63	21	72				
-	28	-	37	6	46	15	55	33	64	22	73				
-	29	-	38	7	47	16	56	34	65	32	74				

NONE (Assistant Examiner)	(Date)	**Total Claims Allowed:** 34	
/Andrew W Johns/ Primary Examiner, Art Unit 2665 (Primary Examiner)	19 May 2021 (Date)	O.G. Print Claim(s) 1	O.G. Print Figure 31

U.S. Patent and Trademark Office Part of Paper No.: 20210519

Search Notes	Application/Control No. 16/584,736	Applicant(s)/Patent Under Reexamination Cosic, Jasmin
	Examiner ANDREW W JOHNS	Art Unit 2665

CPC - Searched*		
Symbol	**Date**	**Examiner**
G06K 9/00355, 9/00389, 9/66	01/29/2021	/AWJ/
G06T 2207/20081	01/29/2021	/AWJ/
G06F 3/002, 3/005, 3/017	01/29/2021	/AWJ/
G06N 5/022, 5/025, 20/00	01/29/2021	/AWJ/
Updated search in symbols listed above	05/14/2021	/AWJ/
Expanded/additional search in symbols listed above	05/17/2021	/AWJ/

CPC Combination Sets - Searched*		
Symbol	**Date**	**Examiner**

US Classification - Searched*			
Class	**Subclass**	**Date**	**Examiner**

* See search history printout included with this form or the SEARCH NOTES box below to determine the scope of the search.

Search Notes		
Search Notes	**Date**	**Examiner**
Reviewed file history and prior art in parent application SN 15/822,150	01/28/2021	/AWJ/
Inventor name search in EAST (search history attached)	01/29/2021	/AWJ/

Search Notes	Application/Control No. 16/584,736	Applicant(s)/Patent Under Reexamination Cosic, Jasmin
	Examiner ANDREW W JOHNS	Art Unit 2665

Interference Search			
US Class/CPC Symbol	**US Subclass/CPC Group**	**Date**	**Examiner**
Interference text search in EAST	(search history attached)	05/19/2021	/AWJ/

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

PTO/SB/08a (01-10)
Approved for use through 07/31/2012. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it contains a valid OMB control number.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS						Remove USPC/CPC
Examiner Initial*	Cite No	Patent Number	Kind Code1	Issue Date	Name of Patentee or Applicant of cited Document	~~Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear~~
/AWJ/	1	8996432	B1	2015-03-31	Fu; Jicheng	G06N 99/005
/AWJ/	2	6754631	B1	2004-06-22	Din	704/270
/AWJ/	3	9305216	B1	2016-04-05	Mishra	G06K 9/00624
/AWJ/	4	5560011	A	1996-09-23	Uyama	395/700
/AWJ/	5	6842877	B2	2005-01-11	Robarts , et al.	715/708
/AWJ/	6	7565340	B2	2009-07-21	Herlocker , et al.	706/52
/AWJ/	7	8261199	B2	2012-09-04	Cradick , et al.	715/757
/AWJ/	8	8266608	B2	2012-09-11	Hecht , et al.	717/158

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

/AWJ/	9	9268454	B2	2016-02-23	Hamilton, II , et al.	G06F 3/0481
/AWJ/	10	8595154	B2	2013-11-26	Breckenridge , et al.	706/12
/AWJ/	11	6408262	B1	2002-06-18	Leerberg , et al.	703/2
/AWJ/	12	9488984	B1	2016-11-08	Williams , et al.	G05D 1/0088
/AWJ/	13	9158813	B2	2015-10-13	Ntoulas , et al.	G06F 17/30451
/AWJ/	14	8356026	B2	2013-01-15	Heimendinger	707/713
/AWJ/	15	10162740	B1	2018-12-25	Setty , et al.	G06F 11/3688

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS Remove

Examiner Initial*	Cite No	Publication Number	Kind Code1	Publication Date	Name of Patentee or Applicant of cited Document	~~Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear~~ USPC/CPC
/AWJ/	1	20120290347	A1	2012-11-15	ELAZOUNI; ASHRAF ; et al.	705/7.12
/AWJ/	2	20150324685	A1	2015-11-12	Bohn; Richard Esten ; et al.	G06N 3/04

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	Application Number		16584736
	Filing Date		2019-09-26
	First Named Inventor	Jasmin Cosic	
	Art Unit		
	Examiner Name		
	Attorney Docket Number		

/AWJ/	3	20100114746	A1	2010-05-06	Bobbitt; Russell Patrick ; et al.	705/35
/AWJ/	4	20120284026	A1	2012-11-08	Cardillo; Peter S. ; et al.	704/246
/AWJ/	5	20140211988	A1	2014-07-31	Fan; Quanfu ; et al.	G06K 9/3241
/AWJ/	6	20150006171	A1	2015-01-01	Westby; Michael C. ; et al.	G10L 15/26
/AWJ/	7	20150264306	A1	2015-09-17	Marilly; Emmanuel ; et al.	H04N 7/147
/AWJ/	8	20110085734	A1	2011-04-14	Berg; Jared S. ; et al.	382/195
/AWJ/	9	20150269415	A1	2015-09-24	Gelbman; Alexander	G06K 9/00214
/AWJ/	10	20160274187	A1	2016-09-22	MENON; SANKARAN M. ; et al.	G01R 31/31718
/AWJ/	11	20150339213	A1	2015-11-22	LEE; Christopher Stephen ; et al.	G06F 11/3664
/AWJ/	12	20150310041	A1	2015-10-29	Kier; Scott ; et al.	G06F 17/30259
/AWJ/	13	20160328480	A1	2016-11-10	Owens; Erich James ; et al.	G06F 17/30867

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

/AWJ/	14	20140075249	A1	2014-03-13	SATO; Shuhei ; et al.	G06F 11/3466
/AWJ/	15	20110270794	A1	2011-11-03	Drory; Tal ; et al.	706/52
/AWJ/	16	20070050719	A1	2007-03-01	Lui; Philip ; et al.	715/762
/AWJ/	17	20070061735	A1	2007-03-15	Hoffberg; Steven M. ; et al.	715/744
/AWJ/	18	20060184410	A1	2006-08-17	Ramamurthy; Shankar ; et al.	705/8
/AWJ/	19	20060265406	A1	2006-11-23	Chkodrov; Gueorgui B. ; et al.	707/100
/AWJ/	20	20090287643	A1	2009-11-19	Corville; Allen O. ; et al.	707/3
/AWJ/	21	20110007079	A1	2011-01-13	Perez et al.	345/473
/AWJ/	22	20070050606	A1	2007-03-01	Ferren et al.	712/227
/AWJ/	23	20090067727	A1	2009-03-12	HIROHATA; Hitoshi	382/201
/AWJ/	24	20130007532	A1	2013-01-03	Miller et al.	714/45

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

/AWJ/	25	20050240412	A1	2005-10-27	Fujita	704/270
/AWJ/	26	20160167226	A1	2016-06-16	Schnittman	B25J 9/163
/AWJ/	27	20090136095	A1	2009-05-28	Marcon et al.	382/118
/AWJ/	28	20080215508	A1	2008-09-04	Hanneman; Jeffrey E. ; et al.	706/11
/AWJ/	29	20090044113	A1	2009-02-12	Jones; Scott T. ; et al.	715/707
/AWJ/	30	20130278501	A1	2013-10-24	BULZACKI; Adrian	G06F 3/017
/AWJ/	31	20190107839	A1	2019-04-11	Parashar; Priyam ; et al.	G05D 1/0219
/AWJ/	32	20180120813	A1	2018-05-03	COFFMAN; Valerie R. ; et al.	G05B 19/4097
/AWJ/	33	20180101146	A1	2018-04-12	Hariharan; Rajaraman ; et al.	G05B 13/026
/AWJ/	34	20160274187	A1	2016-09-22	MENON; SANKARAN M. ; et al.	G01R 31/31718
/AWJ/	35	20140177946	A1	2014-06-26	LIM; Kil-Taek ; et al.	G06K 9/00369

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	Application Number		16584736
	Filing Date		2019-09-26
	First Named Inventor	Jasmin Cosic	
	Art Unit		
	Examiner Name		
	Attorney Docket Number		

/AWJ/	36	20160243701	A1	2016-08-25	Gildert; Suzanne ; et al.	B25J 9/1656
/AWJ/	37	20120167057	A1	2012-06-28	Schmich; Christopher P. ; et al.	717/130
/AWJ/	38	20100278420	A1	2010-11-04	Shet; Vinay Damodar ; et al.	382/156
/AWJ/	39	20170161958	A1	2017-06-08	EILAT; Eran	G06T 19/006
/AWJ/	40	20160077166	A1	2016-03-17	Morozov; Alexey ; et al.	G01R 33/0286
/AWJ/	41	20180068564	A1	2018-03-08	TANIGAWA; TORU ; et al.	G08G 1/146
/AWJ/	42	20120192096	A1	2012-07-26	Bowman; Gordon Gregory ; et al.	715/780
/AWJ/	43	20150269156	A1	2015-09-24	Awadallah; Ahmed Hassan ; et al.	G06F 17/3053
/AWJ/	44	20160063118	A1	2016-03-03	Campbell; Craig S. ; et al.	G06F 17/30867
/AWJ/	45	20140278754	A1	2014-09-18	Cronin; Beau David ; et al.	G06Q 30/0201
/AWJ/	46	20130329946	A1	2013-12-12	ARCHIBALD; Fitzgerald John ; et al.	382/103

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	16584736
	Filing Date	2019-09-26
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number3	Country Code2 i	Kind Code4	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T5
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T5
/AWJ/	1	Chen et al. "Case-Based Reasoning System and Artificial Neural Networks: A Review Neural Comput & Applic (2001) 10: pp 264-276, 13 pages	
/AWJ/	2	JOHN J. GREFENSTETTE, CONNIE LOGGIA RAMSEY, ALAN C. SCHULTZ, Learning Sequential Decision Rules Using Simulation Models and Competition, 1990, Navy Center for Applied Research in Artificial Intelligence, Naval Research Laboratory, Washington, DC, 27 pages	
/AWJ/	3	ALAN C. SCHULTZ, JOHN J. GREFENSTETTE, Using a Genetic Algorithm to Learn Behaviors for Autonomous Vehicles, 1992, Navy Center for Applied Research in Artificial Intelligence, Naval Research Laboratory, Washington, DC, 12 pages	
/AWJ/	4	Koppula et al., "Anticipating human activities using object affordances for reactive robotic response", IEEE TRAMI 2016, published 5 May 2015, 16 pages	
/AWJ/	5	Orme, "System design tips for entry level smartphones - part 3", found online at "https://community.arm.com/processors/b/blog/posts/system-design-tips-for-entry-level-smartphones---part-3", Oct 21, 2013, 12 pages	
/AWJ/	6	NOWOSTAWSKI, MARIUSZ et al., "Dynamic Demes Parallel Genetic Algorithm", May 13, 1999, 6 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	Application Number		16584736
	Filing Date		2019-09-26
	First Named Inventor	Jasmin Cosic	
	Art Unit		
	Examiner Name		
	Attorney Docket Number		

/AWJ/	7		Stack Overflow, How can I quantify difference between two images?, accessed 2 August 2017 at https://stackoverflow.com/questions/189943/how-can-i-quantify-difference-between-two-images, 8 pages	
/AWJ/	8		SKOLICKI, ZBIGNIEW et al., "The Influence of Migration Sizes and Intervals on Island Models", June 29, 2005, 8 pages	
/AWJ/	9		Vahid Lari, et al., "Decentralized dynamic resource management support for massively parallel processor arrays", September 11, 2011, 8 pages	
/AWJ/	10		Vahid Lari, et al., "Distributed resource reservation in massively parallel processor arrays", May 16, 2011, 4 pages	
/AWJ/	11		Mohsen Hayati and Yazdan Shirvany, "Artificial Neural Network Approach for Short Term Load Forecasting for Illam Region", January 2007, 5 pages	
/AWJ/	12		Hasim Sak, Andrew Senior, and Francoise Beaufays, "Long Short-Term Memory Recurrent Neural Network Architectures for Large Scale Acoustic Modeling", January 2014, 5 pages	
/AWJ/	13		Jorg Walter and Klaus Schulten, "Implementation of self-organizing neural networks for visuo-motor control of an industrial robot", January 1993, 10 pages	
/AWJ/	14		Heikki Hyotyniemi and Aarno Lehtola, "A Universal Relation Database Interface for Knowledge Based Systems", April 1991, 5 pages	
/AWJ/	15		Mrissa, Michael, et al. "An avatar architecture for the web of things." IEEE Internet Computing 19.2 (2015): 30-38., 9 pages	
/AWJ/	16		Luck, Michael, and Ruth Aylett. "Applying artificial intelligence to virtual reality: Intelligent virtual environments." Applied Artificial Intelligence 14.1 (2000): 3-32., 30 pages	
/AWJ/	17		Terdjimi, Mehdi, et al. "An avatar-based adaptation workflow for the web of things." Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), 2016 IEEE 25th International Conference on. IEEE, 2016., 6 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	Application Number: 16584736
	Filing Date: 2019-09-26
	First Named Inventor: Jasmin Cosic
	Art Unit:
	Examiner Name:
	Attorney Docket Number:

/AWJ/	18	Bogdanovych, Anton, et al. "Authentic interactive reenactment of cultural heritage with 3D virtual worlds and artificial intelligence." Applied Artificial Intelligence 24.6 (2010): 617-647., 32 pages	
/AWJ/	19	Hernandez, Marco E. Perez, and Stephan Reiff-Marganiec. "Autonomous and self controlling smart objects for the future internet." Future internet of things and cloud (FiCloud), 2015 3rd international conference on. IEEE, 2015., 8 pages	
/AWJ/	20	Medini et al., "Building a Web of Things with Avatars", Managing the Web of Things (2017), 30 pages	
/AWJ/	21	Zhang, Yiying, Lei Guo, and Nicolas D. Georganas. "AGILE: An architecture for agent-based collaborative and interactive virtual environments." Proc. Workshop on Application Virtual Reality Technologies for Future Telecommunication System, IEEE Globecom '2000 Conference. 2000. (Year: 2000), 9 pages	
/AWJ/	22	Eladhari, Mirjam P., "Semi-Autonomous Avatars in Virtual Game Worlds," Pre-conference to the ECREA 2010 - 3rd European Communication Conference, Avatars and Humans. Representing Users in Digital Games, Hamburg, Germany, October 2010, Available from: 2012-01-21 (Year: 2012), 8 pages	

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature	/Andrew W Johns/	Date Considered	04/30/2021

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Inventor:	Jasmin Cosic	Confirmation No.: 3341
Title:	"MACHINE LEARNING FOR COMPUTING ENABLED SYSTEMS AND/OR DEVICES"	
Serial No.:	16/584,736	Filed: September 26, 2019
Examiner:	JOHNS, ANDREW W	Group Art Unit: 2665

Via EFS-Web May 19, 2021

Mail Stop Amendment
COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, VA 22313-1450

AMENDMENT FOR EXAMINER AMENDMENT

Dear Commissioner:

The applicant submits this amendment for Examiner amendment.

Amendments to the Claims begin on page 2.

Remarks begin on page 20.

Listing of Claims

1 - 40 (canceled)

41. (currently amended) A system comprising:

one or more processors; and

one or more non-transitory machine readable media storing machine readable code that, when executed by the one or more processors, causes the one or more processors to perform at least:

receiving or generating a first one or more digital pictures**, wherein the first one or more digital pictures depict at least a portion of a first device's surrounding**;

receiving or generating a first one or more instruction sets for operating **[[a]]** **the** first device; and

learning the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device.

42. (previously presented) The system of claim 41, wherein the receiving the first one or more digital pictures includes receiving the first one or more digital pictures from a picture capturing apparatus, and wherein the receiving the first one or more instruction sets for operating the first device includes receiving the first one or more instruction sets for operating the first device from at least one of: an application for operating the first device, a system for operating the first

device, one or more microcontrollers, another one or more processors, or one or more actuators, and wherein the learning the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device includes storing the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device into or onto at least one of: at least one non-transitory machine readable medium of the one or more non-transitory machine readable media, another one or more non-transitory machine readable media, one or more volatile memories, one or more non-volatile memories, one or more storage devices, or one or more storage systems.

43. (previously presented) The system of claim 41, wherein the learning the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device includes determining that the first one or more instruction sets for operating the first device temporally correspond to the first one or more digital pictures.

44. (previously presented) The system of claim 41, wherein the machine readable code, when executed by the one or more processors, causes the one or more processors to further perform at least:

receiving or generating a new one or more digital pictures;

determining the first one or more instruction sets for operating the first device based on at least partial match between the new one or more digital pictures and the first one or more digital pictures; and

at least in response to the determining, causing the first device or a second device to perform one or more operations defined by the first one or more instruction sets for operating the first device.

45. (previously presented) The system of claim 44, wherein the causing the first device or the second device to perform the one or more operations defined by the first one or more instruction sets for operating the first device includes executing the first one or more instruction sets for operating the first device.

46. (currently amended) The system of claim 44, wherein the determining the first one or more instruction sets for operating the first device based on the at least partial match between the new one or more digital pictures and the first one or more digital pictures includes determining at least partial match between**[[:]]** one or more portions of the new one or more digital pictures that represent one or more ~~**recognized**~~ objects**[[,]]** and one or more portions of the first one or more digital pictures that represent one or more ~~**recognized**~~ objects.

47. (currently amended) The system of claim 44, wherein the determining the first one or more instruction sets for operating the first device based on the at least partial match between the new one or more digital pictures and the first one or more digital pictures includes determining at least partial match between**[[:]]** one or more objects ~~**recognized**~~ **detected** in the new one or more digital

pictures**[[,]]** and one or more objects ~~**recognized**~~ **detected** in the first one or more digital pictures.

48. (previously presented) The system of claim 44, wherein the first one or more instruction sets for operating the first device include one or more information about one or more states of: the first device, or a portion of the first device.

49. (currently amended) The system of claim 44, wherein the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device are included in a knowledgebase, and wherein the knowledgebase further includes a second one or more digital pictures correlated with a second one or more instruction sets for operating the first device, and wherein at least a portion of the first one or more digital pictures or at least a portion of the first one or more instruction sets for operating the first device is learned in a first learning process that includes operating the first device at least partially by a first user, and wherein at least a portion of the second one or more digital pictures or at least a portion of the second one or more instruction sets for operating the first device is learned in a second learning process that includes operating the first device at least partially by the first user~~**, and wherein the first user is: a human user, or a non-human user**~~.

50. (currently amended) The system of claim 44, wherein the first one or more digital pictures correlated with the first one or more instruction sets for operating

the first device are included in a knowledgebase, and wherein the knowledgebase further includes a second one or more digital pictures correlated with a second one or more instruction sets for operating the first device, and wherein at least a portion of the first one or more digital pictures or at least a portion of the first one or more instruction sets for operating the first device is learned in a first learning process that includes operating the first device at least partially by a first user, and wherein at least a portion of the second one or more digital pictures or at least a portion of the second one or more instruction sets for operating the first device is learned in a second learning process that includes operating the first device at least partially by a second user~~**, and wherein the first user is: a human user, or a non-human user, and wherein the second user is: a human user, or a non-human user**~~.

51. (currently amended) The system of claim 44, wherein the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device are included in a knowledgebase, and wherein the knowledgebase further includes a second one or more digital pictures correlated with a second one or more instruction sets for operating a third device, and wherein at least a portion of the first one or more digital pictures or at least a portion of the first one or more instruction sets for operating the first device is learned in a first learning process that includes operating the first device at least partially by a first user, and wherein at least a portion of the second one or more digital pictures or at least a portion of the second one or more instruction sets for

operating the third device is learned in a second learning process that includes operating the third device at least partially by: the first user, or a second user~~,~~ **~~and wherein the first user is: a human user, or a non-human user, and wherein the second user is: a human user, or a non-human user~~**.

52. (currently amended) The system of claim 44, wherein the new one or more digital pictures depict at least a portion of **~~a surrounding of~~** the first device**'s surrounding**, and wherein the first one or more instruction sets for operating the first device are applied to the first device, and wherein the first device is caused to perform the one or more operations defined by the first one or more instruction sets for operating the first device.

53. (currently amended) The system of claim 44, wherein the new one or more digital pictures depict at least a portion of **~~a surrounding of~~** the second **device's surrounding**, and wherein the first one or more instruction sets for operating the first device are applied to the second device, and wherein the second device is caused to perform the one or more operations defined by the first one or more instruction sets for operating the first device.

54. (currently amended) The system of claim 44, wherein the new one or more digital pictures depict at least a portion of **~~a surrounding of~~** the second device**'s surrounding**, and wherein the first one or more instruction sets for operating the first device or a copy of the first one or more instruction sets for operating the first

device are modified and applied to the second device, and wherein the second device is caused to perform one or more operations defined by: the modified the first one or more instruction sets for operating the first device, or the modified the copy of the first one or more instruction sets for operating the first device.

55. (previously presented) The system of claim 44, wherein the machine readable code, when executed by the one or more processors, causes the one or more processors to further perform at least:

modifying:

the first one or more instruction sets for operating the first device, or

a copy of the first one or more instruction sets for operating the first device, and wherein the determining the first one or more instruction sets for operating the first device based on the at least partial match between the new one or more digital pictures and the first one or more digital pictures includes:

determining the modified the first one or more instruction sets for operating the first device based on the at least partial match between the new one or more digital pictures and the first one or more digital pictures, or

determining the modified the copy of the first one or more instruction sets for operating the first device based on the at least partial match between the new one or more digital pictures and the first one or more digital pictures, and wherein the causing the first device or the second device to perform the one or more operations defined by the first one or more instruction sets for operating the first device includes:

causing the first device or the second device to perform one or more operations defined by the modified the first one or more instruction sets for operating the first device, or

causing the first device or the second device to perform one or more operations defined by the modified the copy of the first one or more instruction sets for operating the first device.

56. (currently amended) The system of claim 44, wherein the machine readable code, when executed by the one or more processors, causes the one or more processors to further perform at least:

modifying at least one of: the first one or more digital pictures, a copy of the first one or more digital pictures, the new one or more digital pictures, or a copy of the new one or more digital pictures, and wherein the at least partial match between the new one or more digital pictures and the first one or more digital pictures ~~**is determined at least by**~~ **<u>includes</u>**: (i) ~~**determining**~~ at least partial match between the modified the new one or more digital pictures and the first one or more digital pictures, (ii) ~~**determining**~~ at least partial match between the modified the copy of the new one or more digital pictures and the first one or more digital pictures, (iii) ~~**determining**~~ at least partial match between the new one or more digital pictures and the modified the first one or more digital pictures, (iv) ~~**determining**~~ at least partial match between the new one or more digital pictures and the modified the copy of the first one or more digital pictures, (v) ~~**determining**~~ at least partial match between the modified the new one or more

digital pictures and the modified the first one or more digital pictures, (vi) ~~determining~~ at least partial match between the modified the copy of the new one or more digital pictures and the modified the copy of the first one or more digital pictures, (vii) ~~determining~~ at least partial match between the modified the new one or more digital pictures and the modified the copy of the first one or more digital pictures, or (viii) ~~determining~~ at least partial match between the modified the copy of the new one or more digital pictures and the modified the first one or more digital pictures.

57. (previously presented) The system of claim 44, wherein the machine readable code, when executed by the one or more processors, causes the one or more processors to further perform at least:

modifying:

the first one or more digital pictures, or

a copy of the first one or more digital pictures, and wherein the learning the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device includes:

learning the modified the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device, or

learning the modified the copy of the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device, and wherein the determining the first one or more instruction sets for operating

the first device based on the at least partial match between the new one or more digital pictures and the first one or more digital pictures includes:

determining the first one or more instruction sets for operating the first device based on at least partial match between the new one or more digital pictures and the modified the first one or more digital pictures, or

determining the first one or more instruction sets for operating the first device based on at least partial match between the new one or more digital pictures and the modified the copy of the first one or more digital pictures.

58. (previously presented) The system of claim 44, wherein the machine readable code, when executed by the one or more processors, causes the one or more processors to further perform at least:

modifying:

the first one or more instruction sets for operating the first device, or

a copy of the first one or more instruction sets for operating the first device, and wherein the learning the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device includes:

learning the first one or more digital pictures correlated with the modified the first one or more instruction sets for operating the first device, or

learning the first one or more digital pictures correlated with the modified the copy of the first one or more instruction sets for operating the first device, and wherein the determining the first one or more instruction sets for operating the

first device based on the at least partial match between the new one or more digital pictures and the first one or more digital pictures includes:

determining the modified the first one or more instruction sets for operating the first device based on the at least partial match between the new one or more digital pictures and the first one or more digital pictures, or

determining the modified the copy of the first one or more instruction sets for operating the first device based on the at least partial match between the new one or more digital pictures and the first one or more digital pictures, and wherein the causing the first device or the second device to perform the one or more operations defined by the first one or more instruction sets for operating the first device includes:

causing the first device or the second device to perform one or more operations defined by the modified the first one or more instruction sets for operating the first device, or

causing the first device or the second device to perform one or more operations defined by the modified the copy of the first one or more instruction sets for operating the first device.

59. (previously presented) The system of claim 44, wherein the system further comprising:

an artificial intelligence system, and wherein the learning the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device includes learning, at least in part by the artificial

intelligence system, the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device, and wherein the determining the first one or more instruction sets for operating the first device based on the at least partial match between the new one or more digital pictures and the first one or more digital pictures includes determining, at least in part by the artificial intelligence system, the first one or more instruction sets for operating the first device based on the at least partial match between the new one or more digital pictures and the first one or more digital pictures.

60. (previously presented) The system of claim 44, wherein the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device include the first one or more digital pictures connected, using at least one or more connections, with the first one or more instruction sets for operating the first device.

61. (previously presented) The system of claim 44, wherein the first one or more digital pictures are: one or more whole digital pictures, one or more representations of one or more whole digital pictures, one or more portions of at least one digital picture, one or more representations of one or more portions of at least one digital picture, one or more features, one or more representations of one or more features, one or more collections of pixels, or one or more collections of values, and wherein the new one or more digital pictures are: one or more whole digital pictures, one or more representations of one or more whole

digital pictures, one or more portions of at least one digital picture, one or more representations of one or more portions of at least one digital picture, one or more features, one or more representations of one or more features, one or more collections of pixels, or one or more collections of values.

62. (currently amended) The system of claim 44, wherein the system further comprising:

a server that receives from the first device at least one of: the first one or more digital pictures, or the first one or more instruction sets for operating the first device, and wherein the second device receives from the server **at least one of:** the first one or more digital pictures**, or** ~~**correlated with**~~ the first one or more instruction sets for operating the first device, and wherein the second device is caused to perform the one or more operations defined by the first one or more instruction sets for operating the first device.

63. (previously presented) The system of claim 44, wherein at least some elements of the system are included in: a single device, or multiple devices, and wherein the one or more processors include: one or more microcontrollers, one or more computing devices, or one or more electronic devices, and wherein the first device includes a robot, a vehicle, an appliance, an electronic device, or a mechanical machine, and wherein the second device includes a robot, a vehicle, an appliance, an electronic device, or a mechanical machine, and wherein the first one or more digital pictures include: one or more still digital pictures, or one

or more motion digital pictures, and wherein the new one or more digital pictures include: one or more still digital pictures, or one or more motion digital pictures, and wherein an instruction set of the first one or more instruction sets for operating the first device includes at least one of: only one instruction, a plurality of instructions, one or more inputs, one or more commands, one or more keywords, one or more symbols, one or more operators, one or more variables, one or more parameters, one or more characters, one or more numbers, one or more values, one or more signals, one or more binary bits, one or more functions, one or more function references, one or more objects, one or more object references, one or more data structures, one or more data structure references, one or more states, one or more representations of one or more states, one or more representations of one or more user inputs, one or more codes, one or more data, or one or more information.

64. (currently amended) A method implemented using a computing system that includes one or more processors, the method comprising:

receiving or generating a first one or more digital pictures**, wherein the first one or more digital pictures depict at least a portion of a first device's surrounding**;

receiving or generating a first one or more instruction sets for operating **[[a]]** **the** first device; and

learning the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device.

65. (currently amended) A system comprising:

means for receiving or generating a first one or more digital pictures**,** **wherein the first one or more digital pictures depict at least a portion of a first device's surrounding**;

means for receiving or generating a first one or more instruction sets for operating **[[a]]** **the** first device; and

means for learning the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device.

66. (new) The system of claim 44, wherein the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device are included in a knowledgebase, and wherein the knowledgebase further includes a second one or more digital pictures correlated with a second one or more instruction sets for operating a third device, and wherein the machine readable code, when executed by the one or more processors, causes the one or more processors to further perform at least:

receiving or generating an additional one or more digital pictures;

determining the second one or more instruction sets for operating the third device based on at least partial match between the additional one or more digital pictures and the second one or more digital pictures; and

at least in response to the determining the second one or more instruction sets for operating the third device, causing a fourth device to perform one or

more operations defined by the second one or more instruction sets for operating the third device.

67. (new) The system of claim 44, wherein the at least partial match between the new one or more digital pictures and the first one or more digital pictures is at least in part based on a product of one or more values that represent one or more portions of the new one or more digital pictures and one or more values that represent one or more portions of the first one or more digital pictures.

68. (new) The system of claim 44, wherein the generating the new one or more digital pictures includes:

detecting one or more objects in the first device's surrounding or the second device's surrounding;

generating one or more representations of the one or more objects; and

generating the new one or more digital pictures that include the one or more representations of the one or more objects.

69. (new) The system of claim 68, wherein the detecting the one or more objects in the first device's surrounding or the second device's surrounding includes detecting at least one or more locations of the one or more objects, and wherein the new one or more digital pictures include a top-down view of the one or more representations of the one or more objects.

70. (new) The system of claim 44, wherein the generating the first one or more instruction sets for operating the first device includes generating the first one or more instruction sets for operating the first device as one or more representations of a second one or more instruction sets for operating the first device.

71. (new) The system of claim 59, wherein the artificial intelligence system includes: one or more inputs for receiving one or more digital pictures, and one or more outputs for providing one or more instruction sets, and wherein the learning, at least in part by the artificial intelligence system, the first one or more digital pictures correlated with the first one or more instruction sets for operating the first device includes applying the first one or more digital pictures to the inputs of the artificial intelligence system, and wherein the determining, at least in part by the artificial intelligence system, the first one or more instruction sets for operating the first device based on the at least partial match between the new one or more digital pictures and the first one or more digital pictures includes:

applying the new one or more digital pictures to the inputs of the artificial intelligence system; and

receiving the first one or more instruction sets for operating the first device from the outputs of the artificial intelligence system.

72. (new) The system of claim 71, wherein the one or more inputs for receiving the one or more digital pictures include: one input for one digital picture, multiple

inputs for multiple digital pictures, one input for one portion of one digital picture, multiple inputs for multiple portions of one digital picture, multiple inputs for multiple portions of multiple digital pictures, an input for one representation of one digital picture, multiple inputs for multiple representations of multiple digital pictures, one input for one representation of one portion of one digital picture, multiple inputs for multiple representations of multiple portions of one digital picture, or multiple inputs for multiple representations of multiple portions of multiple digital pictures.

73. (new) The system of claim 59, wherein the artificial intelligence system includes: a knowledgebase, or a neural network.

74. (new) The system of claim 44, wherein the first one or more digital pictures that depict the at least the portion of the first device's surrounding include one or more digital pictures that depict a representation of the at least the portion of the first device's surrounding.

Remarks

Claims 1-40 are canceled. Claims 42-45, 48, 55, 57-61, and 63 are previously presented. Claims 41, 46-47, 49-54, 56, 62, and 64-65 are amended. Claims 66-74 are new. Upon entry of this amendment, claims 41-74 will be pending in the present application. The applicant respectfully requests reconsideration and allowance of all claims.

I. Interview Summary

The applicant thanks the Examiner for conducting a telephonic interview on May 18, 2021 with the applicant.

The Examiner stated that claims presented in the amendment of May 1, 2021 overcome prior art from the Examiner's searches.

The Examiner alleged double patenting related to U.S. Patent No. 9,864,933. The applicant absolutely maintains that the claims as previously presented are allowable and that no double patenting exists. However, to expedite allowance of the application and avoid additional fees, the applicant has amended claim 41 to overcome the alleged double patenting per agreement with the Examiner.

II. Conclusion

The applicant respectfully submits that the amendments presented in this response should not be construed as an admission that the claims as previously presented are not patentable, or that the cited prior art references anticipate or render obvious the claims as previously presented. Indeed, the applicant respectfully submits that the claims as previously presented, and/or further amendments thereto, also contain patentable subject material. Thus, the applicant reserves the right to pursue the claims as previously presented, and/or further amendments thereto, in one or more continuation applications.

In view of the foregoing remarks, the applicant respectfully submits that the entire application is in condition for allowance. The applicant respectfully requests that a timely Notice of Allowance be issued in this case. If the Examiner would like to discuss any aspect of this application, the Examiner is respectfully requested to contact the undersigned at (317) 772-1312.

By /Jasmin Cosic/
Jasmin Cosic
Date submitted: May 19, 2021

Respectfully submitted,
/Jasmin Cosic/
Jasmin Cosic

EAST Search History

EAST Search History (Prior Art)

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
L1	4,240,648	(correlat$5 or associat$5 or link$5 or relat$5) near10 (instruction or control or command or direction)	US-PGPUB; USPAT; USOCR	OR	ON	2021/05/17 07:05
L2	472,405	1 same (image or video or picture or photograph)	US-PGPUB; USPAT; USOCR	OR	ON	2021/05/17 07:07
L3	5,050,917	(device or machine or appliance or equipment) near10 (instruction or control or command or direction)	US-PGPUB; USPAT; USOCR	OR	ON	2021/05/17 07:08
L4	2,066,786	(correlat$5 or associat$5 or link$5 or relat$5) same 3	US-PGPUB; USPAT; USOCR	OR	ON	2021/05/17 07:09
L5	302,590	4 same (image or video or picture or photograph)	US-PGPUB; USPAT; USOCR	OR	ON	2021/05/17 07:10
L6	465,206	(correlat$5 or associat$5 or link$5 or relat$5) near10 (learn$5 or train$5 or teach$5)	US-PGPUB; USPAT; USOCR	OR	ON	2021/05/17 07:11
L7	49,591	6 same (image or video or picture or photograph)	US-PGPUB; USPAT; USOCR	OR	ON	2021/05/17 07:13
L8	35,731	7 and 3	US-PGPUB; USPAT; USOCR	OR	ON	2021/05/17 07:13
L9	3,200	7 same 3	US-PGPUB; USPAT; USOCR	OR	ON	2021/05/17 07:14
L10	7,158	(g06k9/00355 or g06k9/00389 or g06k9/66).cpc.	US-PGPUB; USPAT	OR	ON	2021/05/17 07:14
L11	125	9 and 10	US-PGPUB; USPAT	OR	ON	2021/05/17 07:14
L12	9,128	g06t2207/20081.cpc.	US-PGPUB; USPAT	OR	ON	2021/05/17 07:15
L13	331	9 and 12	US-PGPUB; USPAT	OR	ON	2021/05/17 07:15
L14	310	13 not 11	US-PGPUB; USPAT	OR	ON	2021/05/17 08:20
L15	45,331	(g06n5/022 or g06n5/025 or g06n20/00).cpc.	US-PGPUB; USPAT	OR	ON	2021/05/17 09:24

L16	470	9 and 15	US-PGPUB; USPAT	OR	ON	2021/05/17 09:24
L17	340	16 not (11 or 13)	US-PGPUB; USPAT	OR	ON	2021/05/17 09:24
L18	28,282	(g06f3/002 or g06f3/005 or g06f3/017).cpc.	US-PGPUB; USPAT	OR	ON	2021/05/17 10:36
L19	82	9 and 18	US-PGPUB; USPAT	OR	ON	2021/05/17 10:37

5/17/2021 10:51:15 AM
C:\Users\ajohns\Documents\EAST\Workspaces\Applications\16\500\16584736.wsp

EAST Search History

EAST Search History (Prior Art)

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
L1	4,794,641	(instruction or control$4 or operat$4) near5 device	US-PGPUB; USPAT	OR	ON	2021/05/14 14:01
L2	625,109	(correlat$5 or associat$5) near5 (image or picture or video)	US-PGPUB; USPAT	OR	ON	2021/05/14 14:01
L3	10,876	(learn$4 or train$4) near8 L2	US-PGPUB; USPAT	OR	ON	2021/05/14 14:01
L4	536	L3 same L1	US-PGPUB; USPAT	OR	ON	2021/05/14 14:01
L5	7,158	(g06k9/00355 or g06k9/00389 or g06k9/66).cpc.	US-PGPUB; USPAT	OR	ON	2021/05/14 14:01
L6	42	L4 and L5	US-PGPUB; USPAT	OR	ON	2021/05/14 14:01
L7	9,128	g06t2207/20081.cpc.	US-PGPUB; USPAT	OR	ON	2021/05/14 14:14
L8	95	L4 and L7	US-PGPUB; USPAT	OR	ON	2021/05/14 14:14
L9	28,282	(g06f3/002 or g06f3/005 or g06f3/017).cpc.	US-PGPUB; USPAT	OR	ON	2021/05/14 14:45
L10	4	L4 and L9	US-PGPUB; USPAT	OR	ON	2021/05/14 14:45
L11	45,331	(g06n5/022 or g06n5/025 or g06n20/00).cpc.	US-PGPUB; USPAT	OR	ON	2021/05/14 14:45
L12	86	L4 and L11	US-PGPUB; USPAT	OR	ON	2021/05/14 14:45

5/14/2021 3:03:47 PM
C:\Users\ajohns\Documents\EAST\Workspaces\Applications\16\500\16584736.wsp

Bibliographic Data

Application No: 16/584,736

Foreign Priority claimed: ○ Yes ◉ No

35 USC 119 (a-d) conditions met: ☐ Yes ☑ No ☐ Met After Allowance

Verified and Acknowledged: /Andrew W Johns/

Examiner's Signature Initials

Title: MACHINE LEARNING FOR COMPUTING ENABLED SYSTEMS AND/OR DEVICES

FILING or 371(c) DATE	CLASS	GROUP ART UNIT	ATTORNEY DOCKET NO.
09/26/2019	382	2665	
RULE			

APPLICANTS

INVENTORS

Jasmin Cosic, Miami, FL, UNITED STATES

CONTINUING DATA

This application is a CON of 15822150 11/26/2017 PAT 10474934

FOREIGN APPLICATIONS

IF REQUIRED, FOREIGN LICENSE GRANTED**

10/09/2019

**** SMALL ENTITY ****

STATE OR COUNTRY

UNITED STATES

ADDRESS

Jasmin Cosic
108 Woodbury Street
Pawtucket, RI 02861
UNITED STATES

FILING FEE RECEIVED

$985

EAST Search History

EAST Search History (Interference)

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
L1	894,387	((device or machine or appliance or equipment) near10 (instruction or control or command or direction)).clm.	US-PGPUB	OR	ON	2021/05/19 06:39
L2	443,025	((correlat$5 or associat$5 or link$5 or relat$5) near10 (instruction or control or command or direction)).clm.	US-PGPUB	OR	ON	2021/05/19 06:40
L3	25,428	((correlat$5 or associat$5 or link$5 or relat$5) near10 (learn$5 or train$5 or teach$5)).clm.	US-PGPUB	OR	ON	2021/05/19 06:42
L4	3,372	3 same (image or video or picture or photograph).clm.	US-PGPUB	OR	ON	2021/05/19 06:43
L5	303	2 same 4	US-PGPUB	OR	ON	2021/05/19 06:43
L6	115	5 same 1	US-PGPUB	OR	ON	2021/05/19 06:43
L7	756,588	((receiv$5 or input$5) near10 (instruction or control or command or direction)).clm.	US-PGPUB	OR	ON	2021/05/19 06:44
L8	77	5 same 7	US-PGPUB	OR	ON	2021/05/19 06:45
L9	45	8 and 1	US-PGPUB	OR	ON	2021/05/19 06:45

5/19/2021 6:56:03 AM
C:\Users\ajohns\Documents\EAST\Workspaces\Applications\16\500\16584736.wsp

PART B - FEE(S) TRANSMITTAL

Complete and send this form, together with applicable fee(s), by mail or fax, or via EFS-Web.

By mail, send to: Mail Stop ISSUE FEE
Commissioner for Patents
P.O. Box 1450
Alexandria, Virginia 22313-1450

By fax, send to: (571)-273-2885

INSTRUCTIONS: This form should be used for transmitting the ISSUE FEE and PUBLICATION FEE (if required). Blocks 1 through 5 should be completed where appropriate. All further correspondence including the Patent, advance orders and notification of maintenance fees will be mailed to the current correspondence address as indicated unless corrected below or directed otherwise in Block 1, by (a) specifying a new correspondence address; and/or (b) indicating a separate "FEE ADDRESS" for maintenance fee notifications.

CURRENT CORRESPONDENCE ADDRESS (Note: Use Block 1 for any change of address)

116094 7590 05/25/2021

Jasmin Cosic
108 Woodbury Street
Pawtucket, RI 02861

Note: A certificate of mailing can only be used for domestic mailings of the Fee(s) Transmittal. This certificate cannot be used for any other accompanying papers. Each additional paper, such as an assignment or formal drawing, must have its own certificate of mailing or transmission.

Certificate of Mailing or Transmission

I hereby certify that this Fee(s) Transmittal is being deposited with the United States Postal Service with sufficient postage for first class mail in an envelope addressed to the Mail Stop ISSUE FEE address above, or being transmitted to the USPTO via EFS-Web or by facsimile to (571) 273-2885, on the date below.

Jasmin Cosic	(Typed or printed name)
/Jasmin Cosic/	(Signature)
5.27.2021	(Date)

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
16/584,736	09/26/2019	Jasmin Cosic		3341

TITLE OF INVENTION: MACHINE LEARNING FOR COMPUTING ENABLED SYSTEMS AND/OR DEVICES

APPLN. TYPE	ENTITY STATUS	ISSUE FEE DUE	PUBLICATION FEE DUE	PREV. PAID ISSUE FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	SMALL	$600	$0.00	$0.00	$600	08/25/2021

EXAMINER	ART UNIT	CLASS-SUBCLASS
JOHNS, ANDREW W	2665	382-157000

1. Change of correspondence address or indication of "Fee Address" (37 CFR 1.363).

☐ Change of correspondence address (or Change of Correspondence Address form PTO/SB/122) attached.

☐ "Fee Address" indication (or "Fee Address" Indication form PTO/SB/47; Rev 03-09 or more recent) attached. **Use of a Customer Number is required.**

2. For printing on the patent front page, list
(1) The names of up to 3 registered patent attorneys or agents OR, alternatively,
(2) The name of a single firm (having as a member a registered attorney or agent) and the names of up to 2 registered patent attorneys or agents. If no name is listed, no name will be printed.

1 ______
2 ______
3 ______

3. ASSIGNEE NAME AND RESIDENCE DATA TO BE PRINTED ON THE PATENT (print or type)

PLEASE NOTE: Unless an assignee is identified below, no assignee data will appear on the patent. If an assignee is identified below, the document must have been previously recorded, or filed for recordation, as set forth in 37 CFR 3.11 and 37 CFR 3.81(a). Completion of this form is NOT a substitute for filing an assignment.

(A) NAME OF ASSIGNEE

(B) RESIDENCE: (CITY and STATE OR COUNTRY)

Please check the appropriate assignee category or categories (will not be printed on the patent) : ☐ Individual ☐ Corporation or other private group entity ☐ Government

4a. Fees submitted: ☒ Issue Fee ☐ Publication Fee (if required) ☐ Advance Order - # of Copies ______

4b. Method of Payment: *(Please first reapply any previously paid fee shown above)*

☒ Electronic Payment via EFS-Web ☐ Enclosed check ☐ Non-electronic payment by credit card (Attach form PTO-2038)

☐ The Director is hereby authorized to charge the required fee(s), any deficiency, or credit any overpayment to Deposit Account No. ______

5. **Change in Entity Status** (from status indicated above)

☐ Applicant certifying micro entity status. See 37 CFR 1.29

☐ Applicant asserting small entity status. See 37 CFR 1.27

☐ Applicant changing to regular undiscounted fee status.

NOTE: Absent a valid certification of Micro Entity Status (see forms PTO/SB/15A and 15B), issue fee payment in the micro entity amount will not be accepted at the risk of application abandonment.
NOTE: If the application was previously under micro entity status, checking this box will be taken to be a notification of loss of entitlement to micro entity status.
NOTE: Checking this box will be taken to be a notification of loss of entitlement to small or micro entity status, as applicable.

NOTE: This form must be signed in accordance with 37 CFR 1.31 and 1.33. See 37 CFR 1.4 for signature requirements and certifications.

Authorized Signature /Jasmin Cosic/ Date 5.27.2021

Typed or printed name Jasmin Cosic Registration No. ______

PTOL-85 Part B (08-18) Approved for use through 01/31/2020 OMB 0651-0033 U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

Electronic Patent Application Fee Transmittal

Application Number:	16584736
Filing Date:	26-Sep-2019
Title of Invention:	MACHINE LEARNING FOR COMPUTING ENABLED SYSTEMS AND/OR DEVICES
First Named Inventor/Applicant Name:	Jasmin Cosic
Filer:	Jasmin Cosic
Attorney Docket Number:	

Filed as Small Entity

Filing Fees for **Utility under 35 USC 111(a)**

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Basic Filing:				
Pages:				
Claims:				
Miscellaneous-Filing:				
Petition:				
Patent-Appeals-and-Interference:				
Post-Allowance-and-Post-Issuance:				
UTILITY APPL ISSUE FEE	2501	1	600	600

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Extension-of-Time:				
Miscellaneous:				
	Total in USD ($)			**600**

Electronic Acknowledgement Receipt

EFS ID:	42841597
Application Number:	16584736
International Application Number:	
Confirmation Number:	3341
Title of Invention:	MACHINE LEARNING FOR COMPUTING ENABLED SYSTEMS AND/OR DEVICES
First Named Inventor/Applicant Name:	Jasmin Cosic
Customer Number:	116094
Filer:	Jasmin Cosic
Filer Authorized By:	
Attorney Docket Number:	
Receipt Date:	27-MAY-2021
Filing Date:	26-SEP-2019
Time Stamp:	17:06:16
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	yes
Payment Type	CARD
Payment was successfully received in RAM	$600
RAM confirmation Number	E20215QH08002791
Deposit Account	
Authorized User	

The Director of the USPTO is hereby authorized to charge indicated fees and credit any overpayment as follows:

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1	Issue Fee Payment (PTO-85B)	PTOL-85B_Issue_Fee_Form.pdf	110150 aa591844a8b4b5e1e7266a3c914ecd5bce6b3928	no	1
Warnings:					
Information:					
2	Fee Worksheet (SB06)	fee-info.pdf	30223 231cc02dc7c7266294fab93d71b71a5b7dfaa75c	no	2
Warnings:					
Information:					
		Total Files Size (in bytes):	140373		

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

<u>New Applications Under 35 U.S.C. 111</u>
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.
<u>National Stage of an International Application under 35 U.S.C. 371</u>
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.
<u>New International Application Filed with the USPTO as a Receiving Office</u>
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	ISSUE DATE	PATENT NO.	ATTORNEY DOCKET NO.	CONFIRMATION NO.
16/584,736	07/06/2021	11055583		3341

116094 7590 06/16/2021

Jasmin Cosic
108 Woodbury Street
Pawtucket, RI 02861

ISSUE NOTIFICATION

The projected patent number and issue date are specified above.

Determination of Patent Term Adjustment under 35 U.S.C. 154 (b)
(application filed on or after May 29, 2000)

The Patent Term Adjustment is 67 day(s). Any patent to issue from the above-identified application will include an indication of the adjustment on the front page.

If a Continued Prosecution Application (CPA) was filed in the above-identified application, the filing date that determines Patent Term Adjustment is the filing date of the most recent CPA.

Applicant will be able to obtain more detailed information by accessing the Patent Application Information Retrieval (PAIR) WEB site (http://pair.uspto.gov).

Any questions regarding the Patent Term Extension or Adjustment determination should be directed to the Office of Patent Legal Administration at (571)-272-7702. Questions relating to issue and publication fee payments should be directed to the Application Assistance Unit (AAU) of the Office of Data Management (ODM) at (571)-272-4200.

APPLICANT(s) (Please see PAIR WEB site http://pair.uspto.gov for additional applicants):

Jasmin Cosic, Miami, FL;

Exhibit O

PTO/AIA/01 (06-12)
Approved for use through 01/31/2014. OMB 0651-0032
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

DECLARATION (37 CFR 1.63) FOR UTILITY OR DESIGN APPLICATION USING AN APPLICATION DATA SHEET (37 CFR 1.76)

Title of Invention	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING AN AVATAR'S CIRCUMSTANCES FOR AUTONOMOUS AVATAR OPERATION

As the below named inventor, I hereby declare that:

This declaration is directed to:

- [x] The attached application, or
- [ ] United States application or PCT international application number ____________________

filed on ____________________.

The above-identified application was made or authorized to be made by me.

I believe that I am the original inventor or an original joint inventor of a claimed invention in the application.

I hereby acknowledge that any willful false statement made in this declaration is punishable under 18 U.S.C. 1001 by fine or imprisonment of not more than five (5) years, or both.

WARNING:

Petitioner/applicant is cautioned to avoid submitting personal information in documents filed in a patent application that may contribute to identity theft. Personal information such as social security numbers, bank account numbers, or credit card numbers (other than a check or credit card authorization form PTO-2038 submitted for payment purposes) is never required by the USPTO to support a petition or an application. If this type of personal information is included in documents submitted to the USPTO, petitioners/applicants should consider redacting such personal information from the documents before submitting them to the USPTO. Petitioner/applicant is advised that the record of a patent application is available to the public after publication of the application (unless a non-publication request in compliance with 37 CFR 1.213(a) is made in the application) or issuance of a patent. Furthermore, the record from an abandoned application may also be available to the public if the application is referenced in a published application or an issued patent (see 37 CFR 1.14). Checks and credit card authorization forms PTO-2038 submitted for payment purposes are not retained in the application file and therefore are not publicly available.

LEGAL NAME OF INVENTOR

Inventor: Jasmin Cosic Date (Optional) : ____________________

Signature: ____________________

Note: An application data sheet (PTO/SB/14 or equivalent), including naming the entire inventive entity, must accompany this form or must have been previously filed. Use an additional PTO/AIA/01 form for each additional inventor.

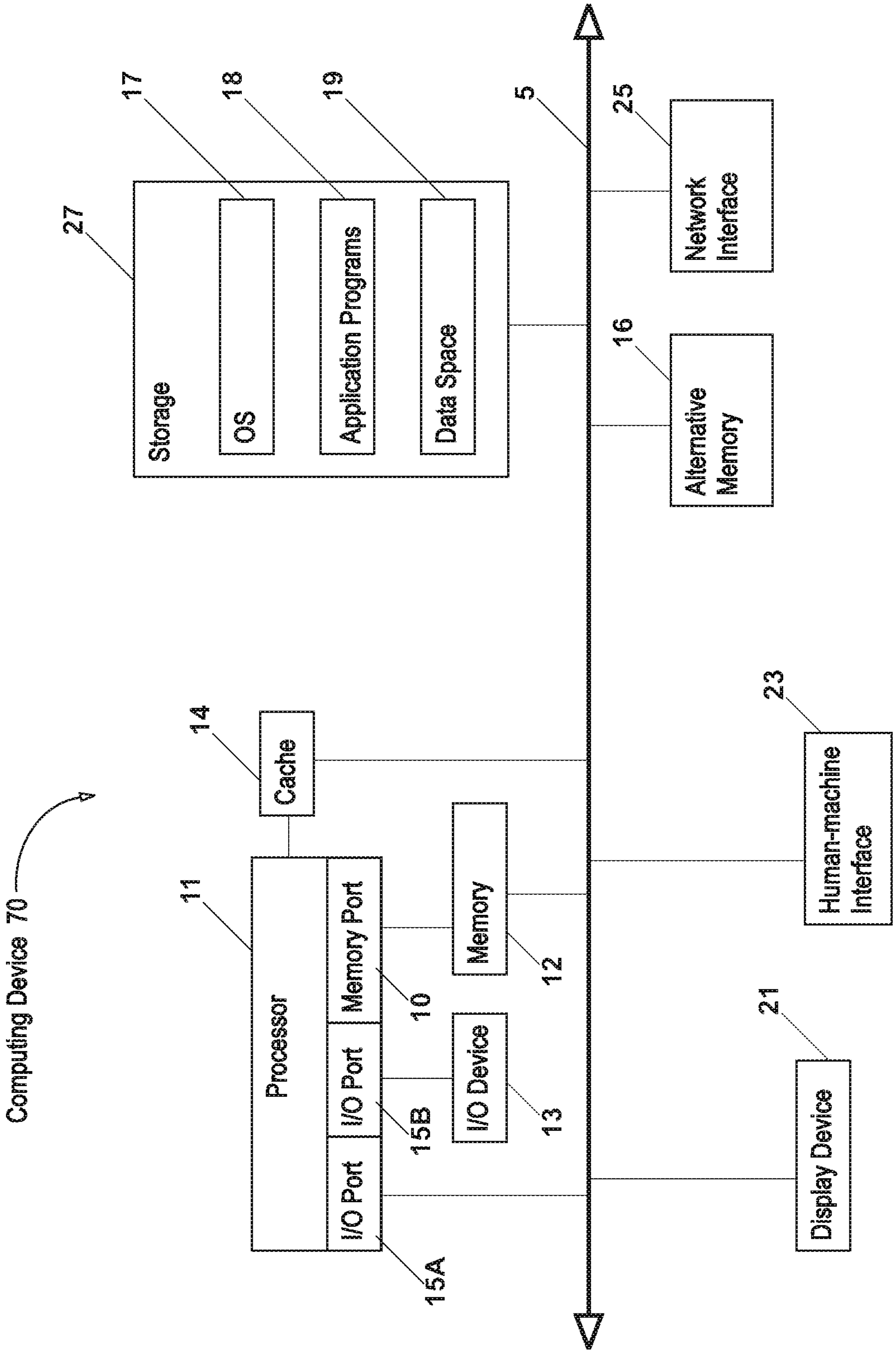

Computing Device 70
Processor
11
I/O Port
I/O Port
Memory Port
15A
15B
10
Cache
14
Memory
12
I/O Device
13
Storage
OS
Application Programs
Data Space
27
17
18
19
5
Network Interface
25
Alternative Memory
16
Human-machine Interface
23
Display Device
21

FIG. 1

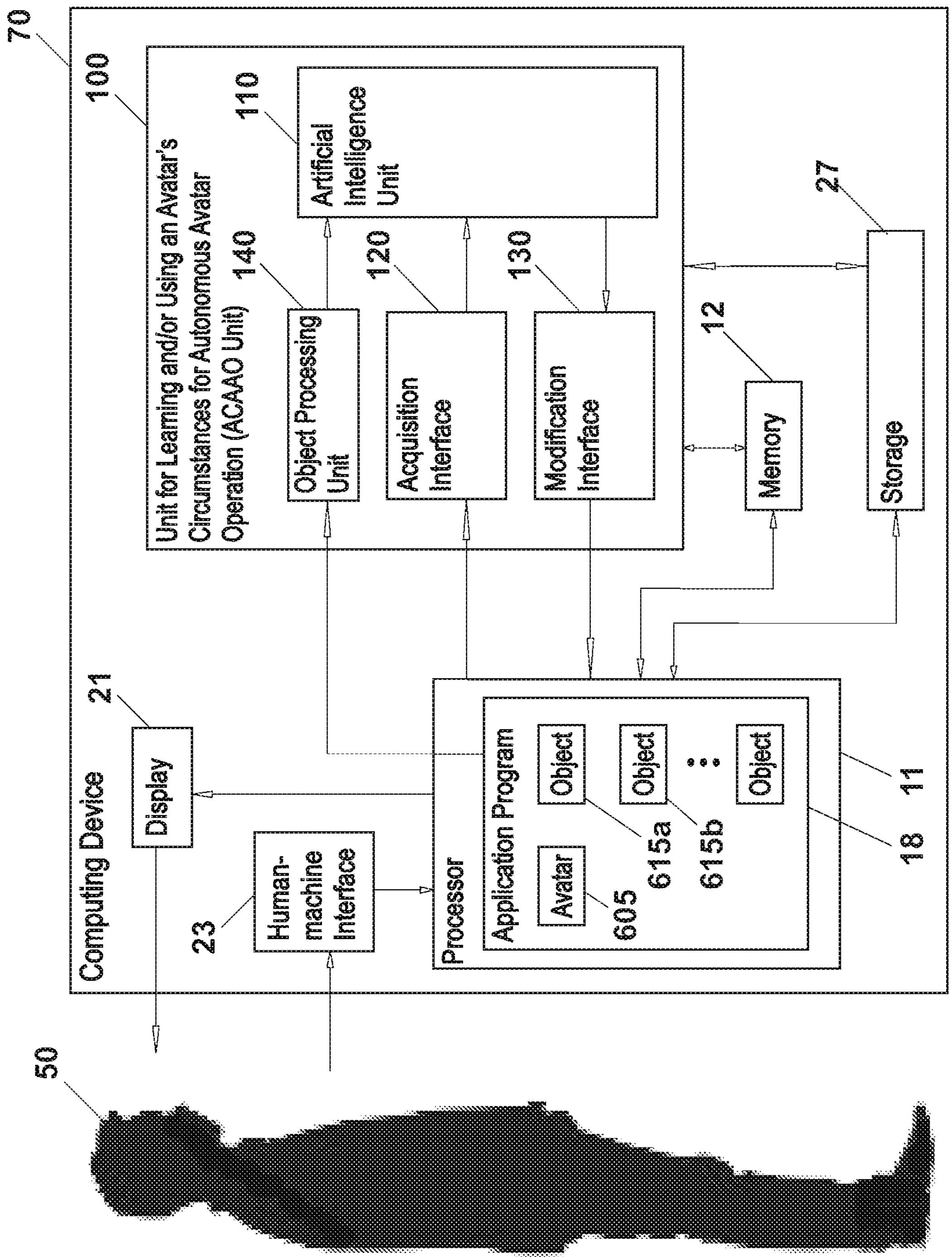

70
100
Unit for Learning and/or Using an Avatar's Circumstances for Autonomous Avatar Operation (ACAAO Unit)
110
Artificial Intelligence Unit
140
Object Processing Unit
120
Acquisition Interface
130
Modification Interface
27
12
Memory
Storage
Computing Device
21
Display
23
Human-machine Interface
Processor
Application Program
Avatar
605
Object
615a
Object
615b
Object
11
18
50

FIG. 2

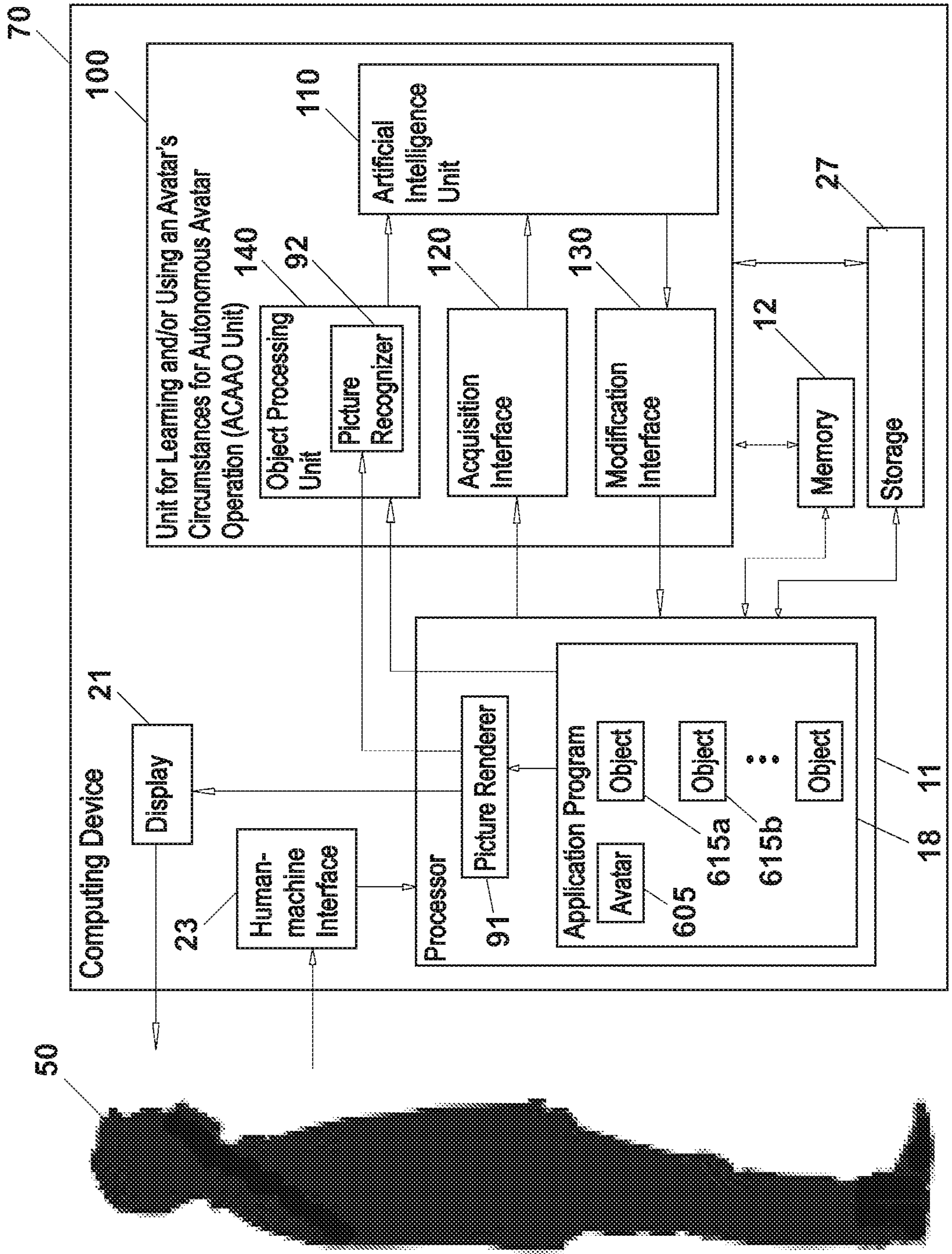

70
Computing Device
100
Unit for Learning and/or Using an Avatar's Circumstances for Autonomous Avatar Operation (ACAAO Unit)
110
Artificial Intelligence Unit
140
92
Object Processing Unit
Picture Recognizer
120
Acquisition Interface
130
Modification Interface
12
Memory
27
Storage
21
Display
23
Human-machine Interface
Processor
91
Picture Renderer
Application Program
Avatar
605
Object
615a
Object
615b
Object
11
18
50

FIG. 3

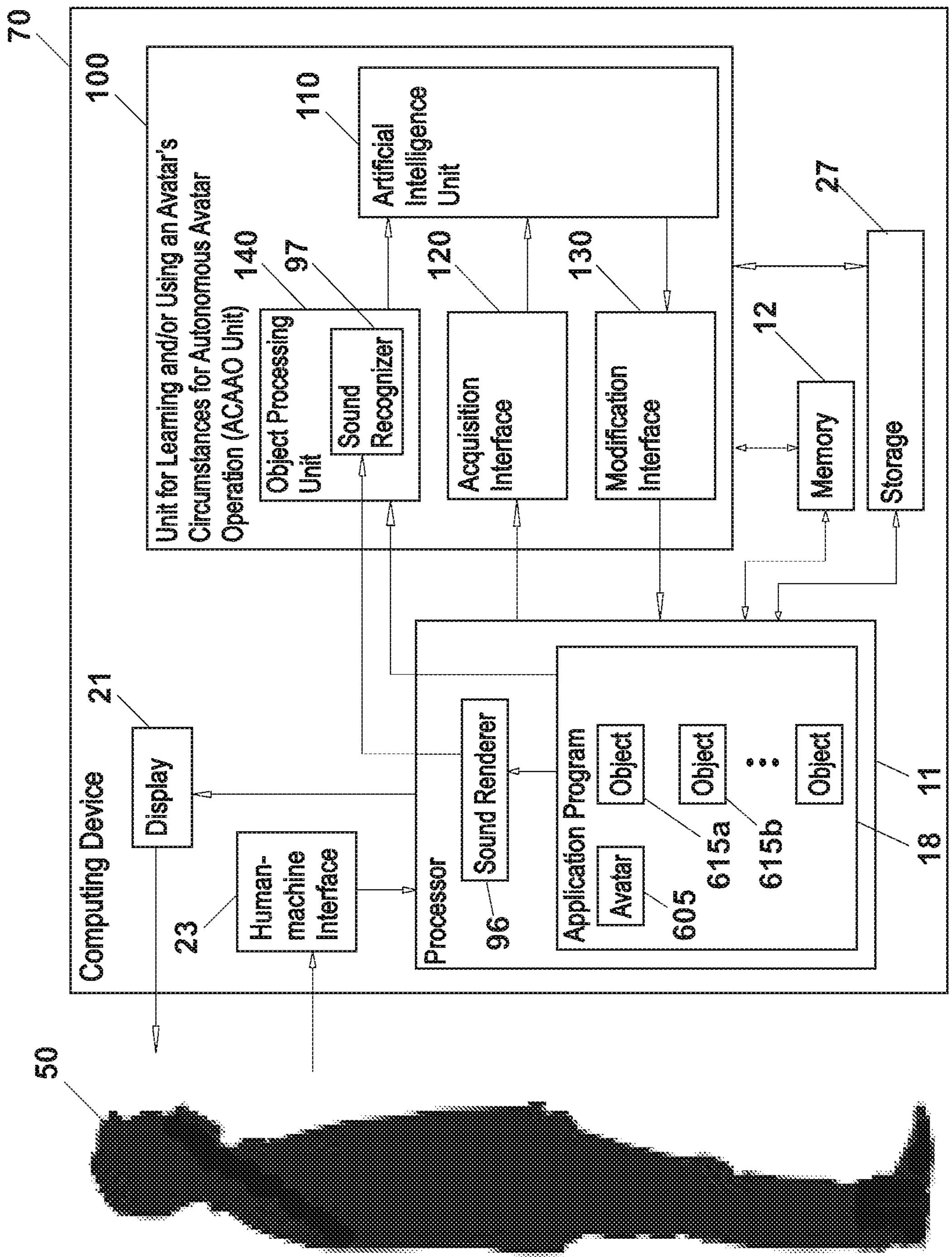

70
Computing Device
100
Unit for Learning and/or Using an Avatar's Circumstances for Autonomous Avatar Operation (ACAAO Unit)
110
Artificial Intelligence Unit
140
97
Object Processing Unit
Sound Recognizer
120
Acquisition Interface
130
Modification Interface
12
Memory
27
Storage
21
Display
23
Human-machine Interface
Processor
96
Sound Renderer
Application Program
Avatar
605
Object
615a
Object
615b
Object
18
11
50

FIG. 4

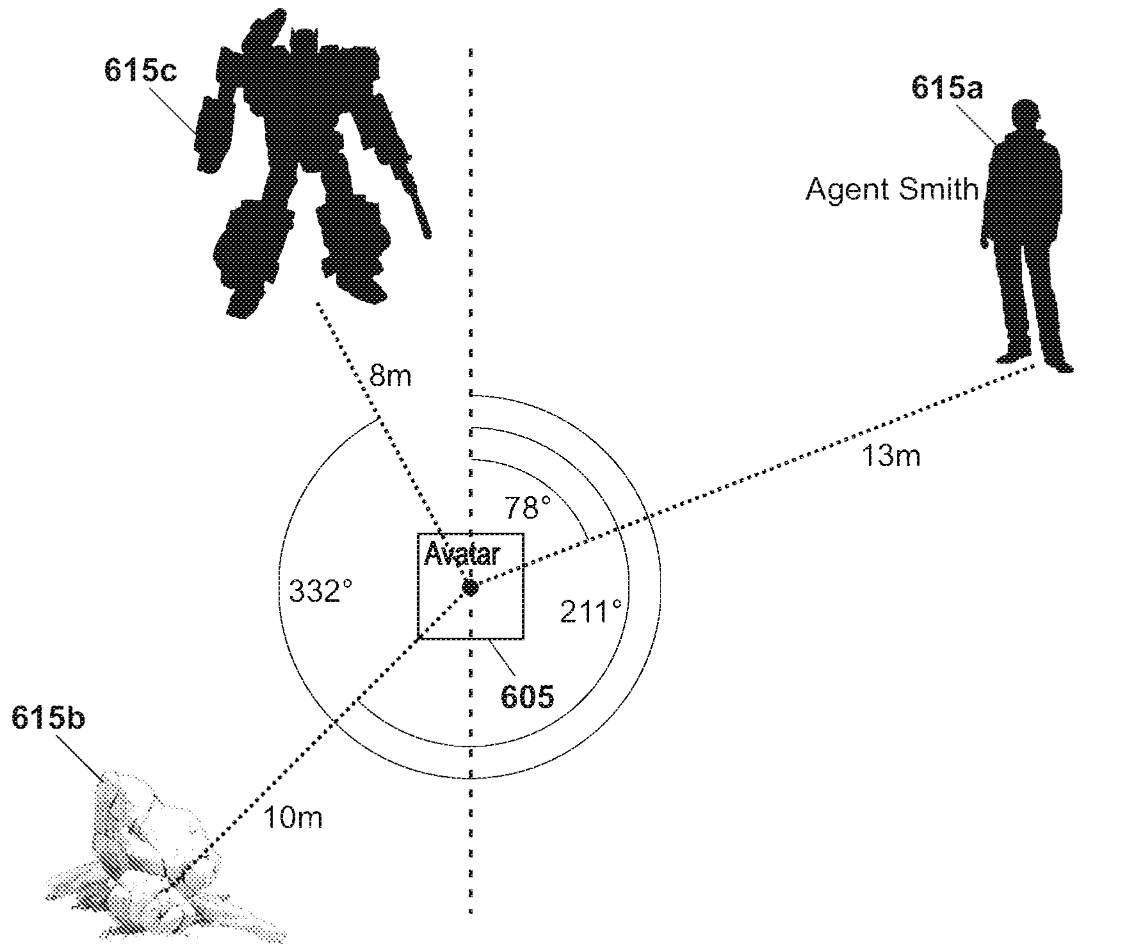

615c
615a
Agent Smith
8m
13m
78°
Avatar
332°
211°
605
615b
10m

FIG. 5A

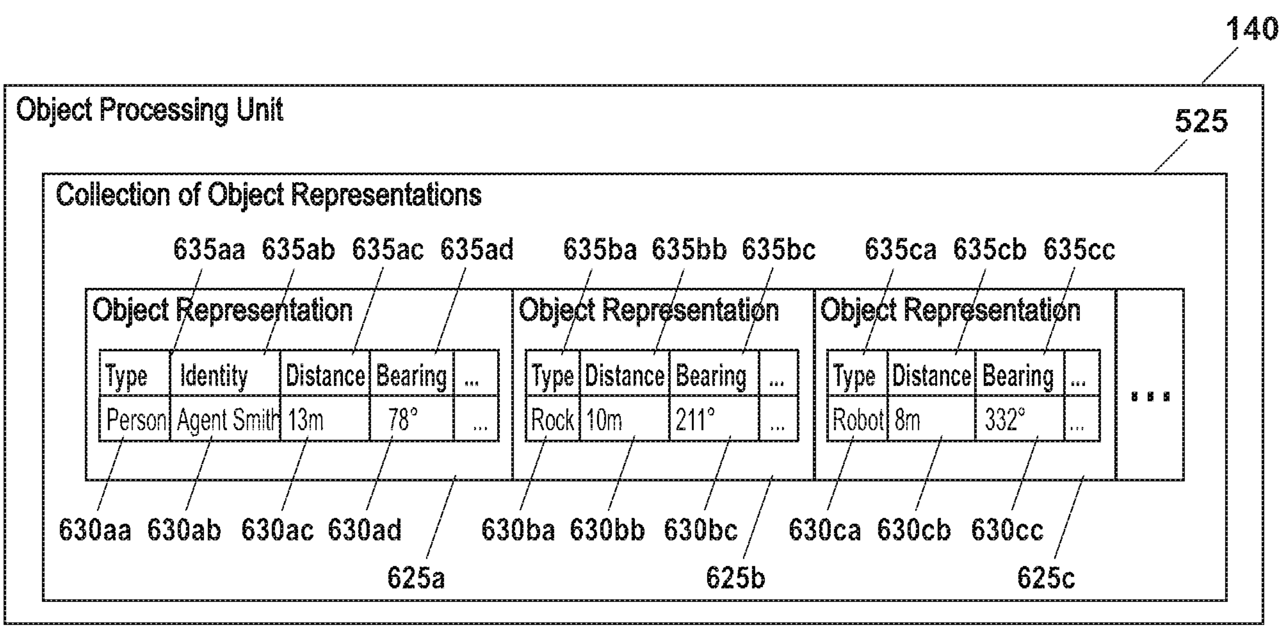

140
Object Processing Unit
525
Collection of Object Representations
635aa 635ab 635ac 635ad
635ba 635bb 635bc
635ca 635cb 635cc
Object Representation
Type | Identity | Distance | Bearing | ...
Person | Agent Smith | 13m | 78° | ...
Object Representation
Type | Distance | Bearing | ...
Rock | 10m | 211° | ...
Object Representation
Type | Distance | Bearing | ...
Robot | 8m | 332° | ...
...
630aa 630ab 630ac 630ad
630ba 630bb 630bc
630ca 630cb 630cc
625a
625b
625c

FIG. 5B

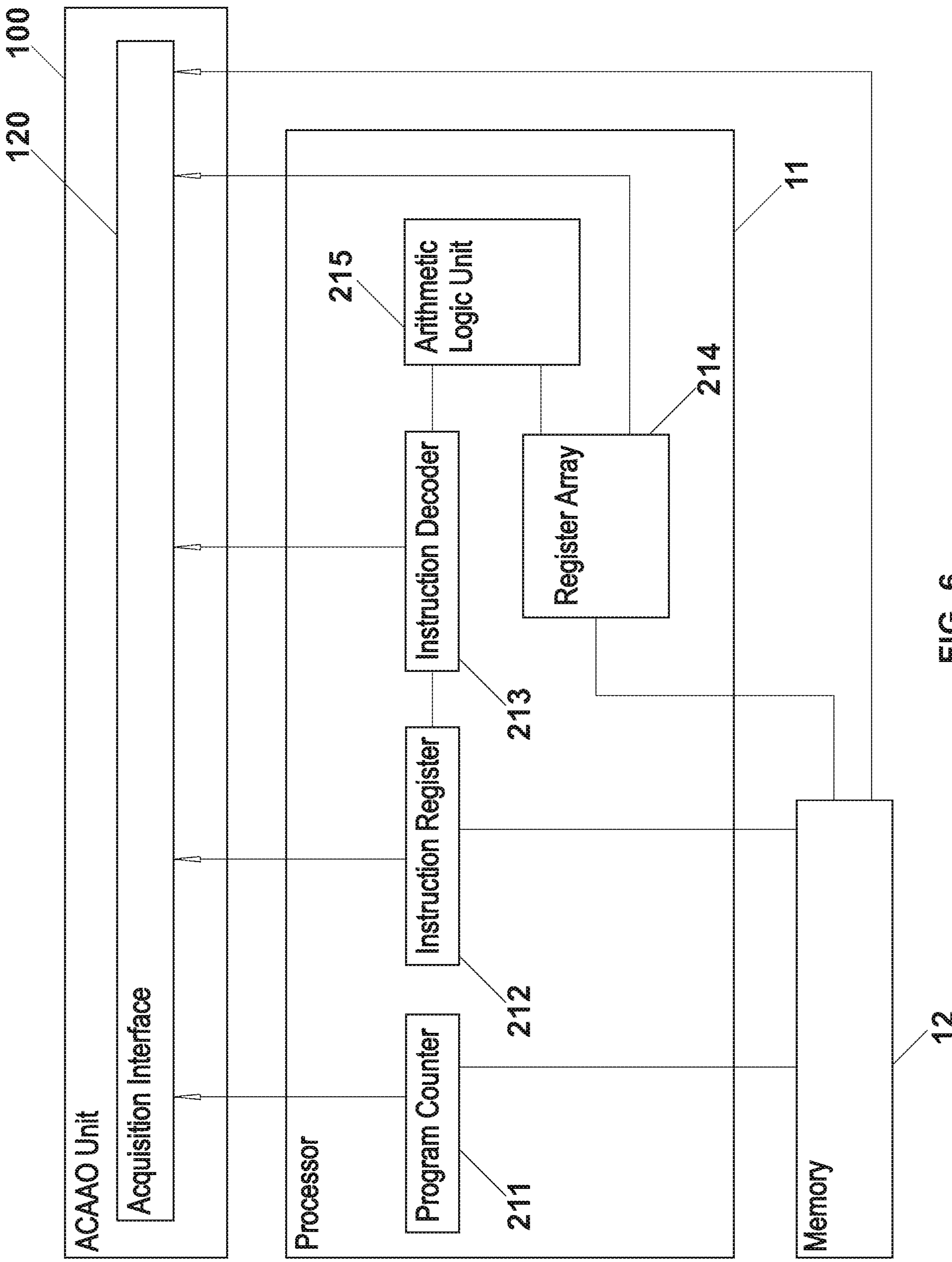

100
120
ACAAO Unit
Acquisition Interface
Processor
11
215
Arithmetic Logic Unit
214
Instruction Decoder
Register Array
213
Instruction Register
212
Program Counter
211
Memory
12

FIG. 6

526

Instruction Set

Function (Parameter1, Parameter2, Parameter3, ...)

FIG. 7A

526

Instruction Set

SELECT Col1, Col2 ... FROM Tbl1, Tbl2 ... ;

FIG. 7B

526

Instruction Set

iconst_Value

FIG. 7C

526

Instruction Set

add Register1 Register2

FIG. 7D

526

Instruction Set

000000 00001 00010 00110 00000 100000

FIG. 7E

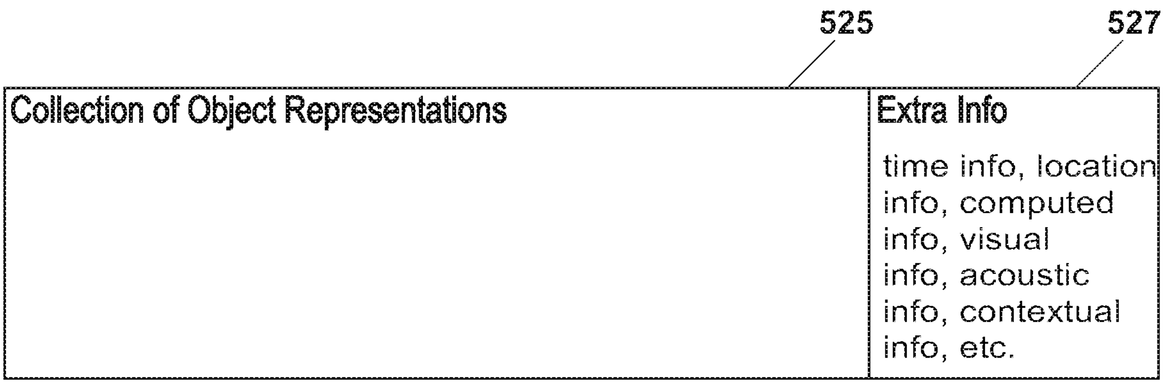

525
527
Collection of Object Representations
Extra Info
time info, location info, computed info, visual info, acoustic info, contextual info, etc.

FIG. 8A

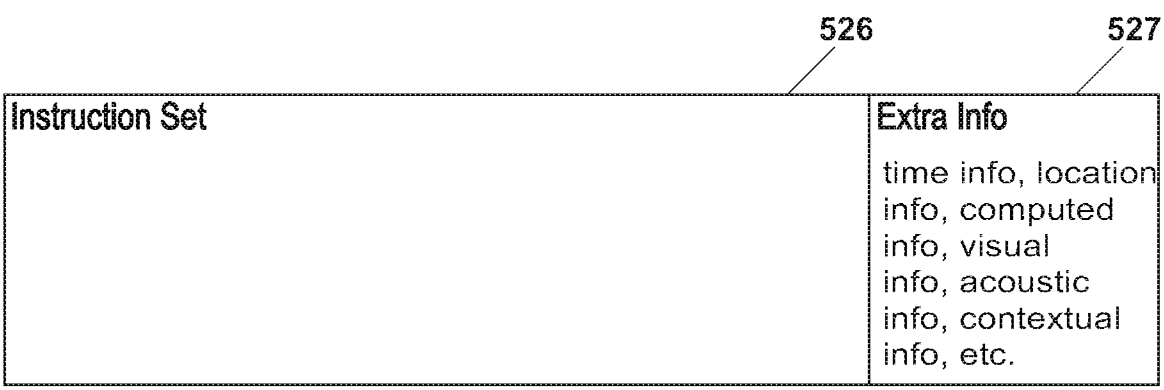

526
527
Instruction Set
Extra Info
time info, location info, computed info, visual info, acoustic info, contextual info, etc.

FIG. 8B

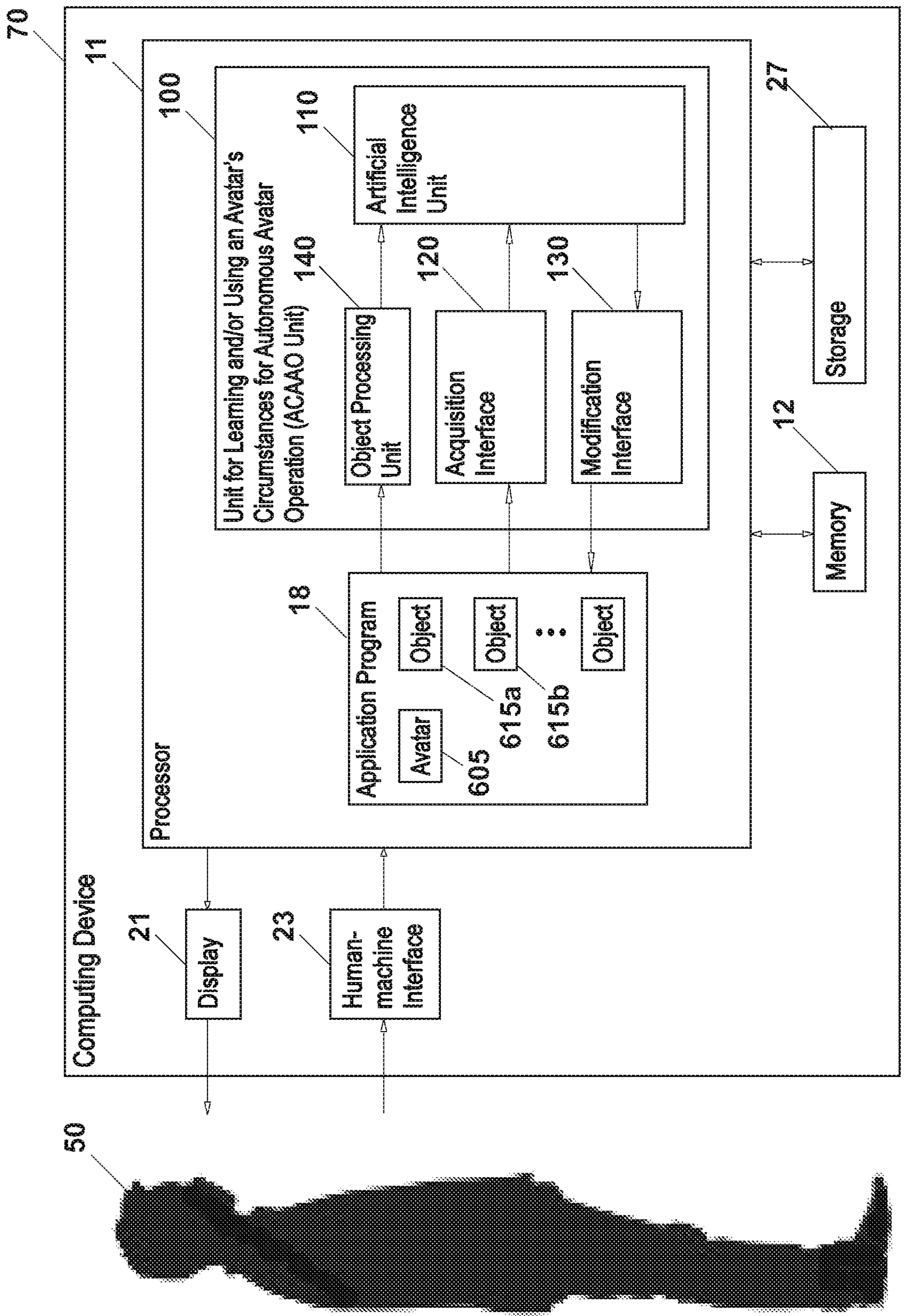

70
Computing Device
11
Processor
100
Unit for Learning and/or Using an Avatar's Circumstances for Autonomous Avatar Operation (ACAAO Unit)
110
Artificial Intelligence Unit
140
Object Processing Unit
120
Acquisition Interface
130
Modification Interface
18
Application Program
Avatar
605
Object
615a
Object
615b
Object
21
Display
23
Human-machine Interface
12
Memory
27
Storage
50

FIG. 9

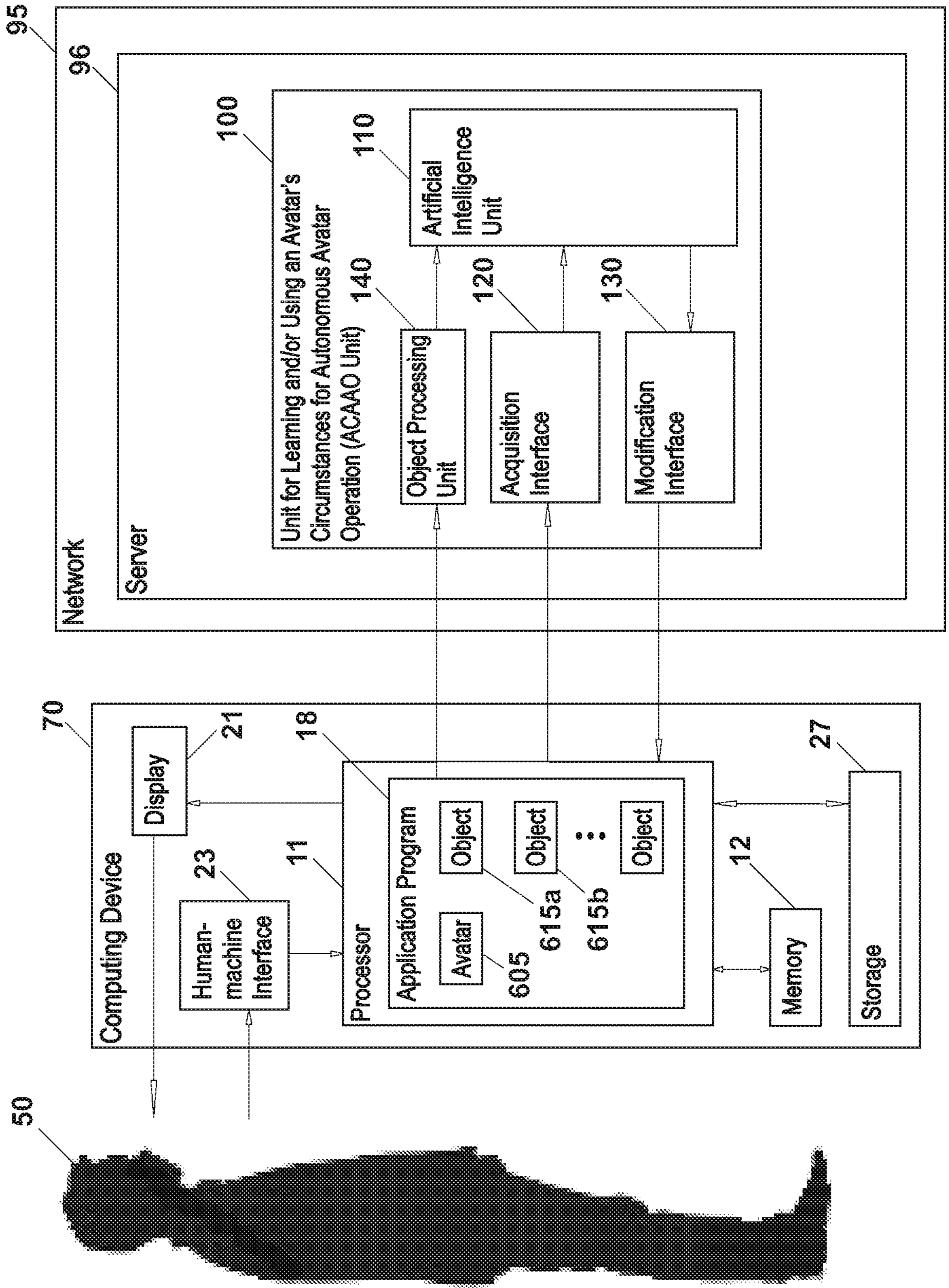

95
96
Network
Server
100
Unit for Learning and/or Using an Avatar's Circumstances for Autonomous Avatar Operation (ACAAO Unit)
110
Artificial Intelligence Unit
140
Object Processing Unit
120
Acquisition Interface
130
Modification Interface
70
Computing Device
21
Display
23
Human-machine Interface
11
Processor
18
Application Program
Avatar
605
Object
615a
Object
615b
Object
12
Memory
27
Storage
50

FIG. 10

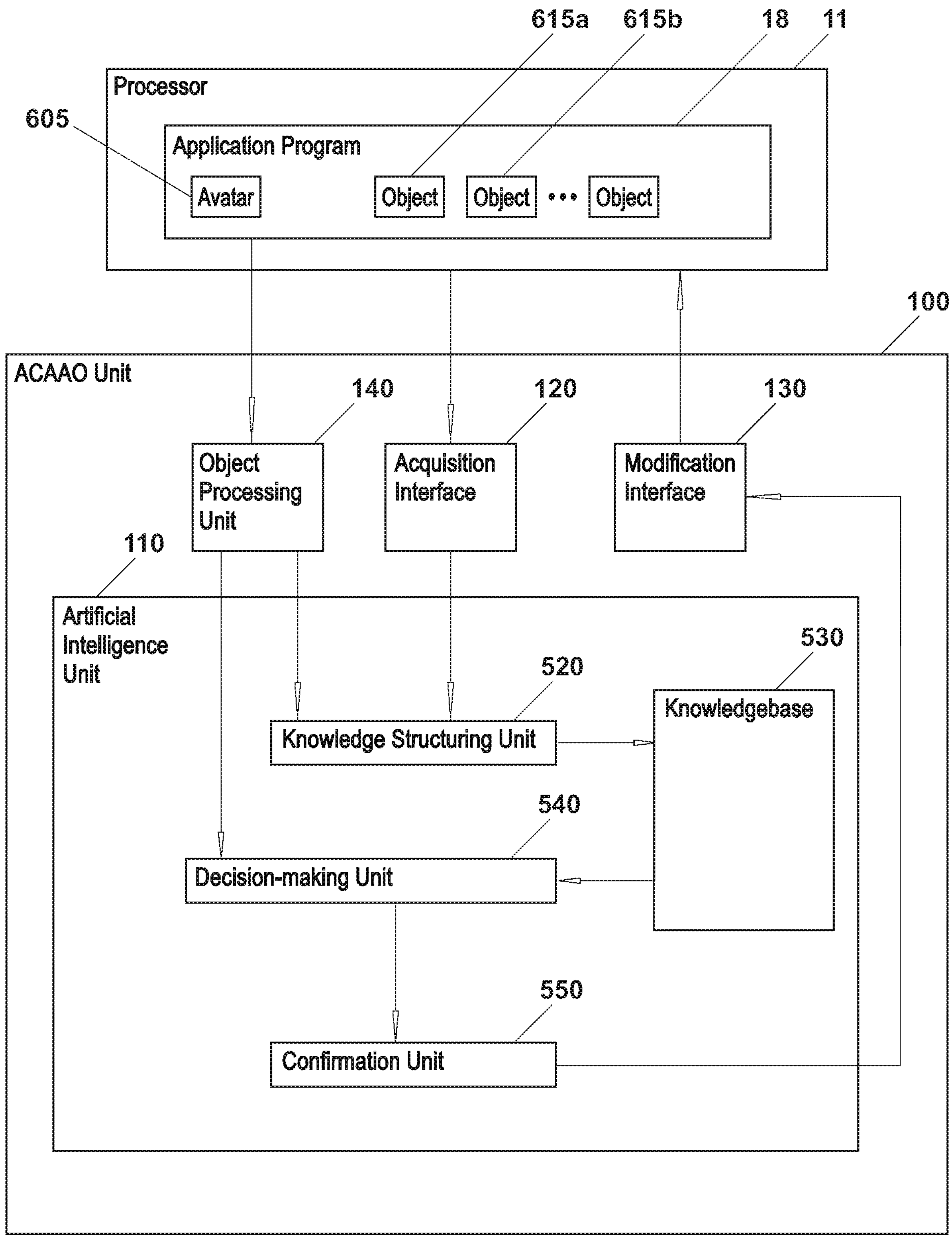
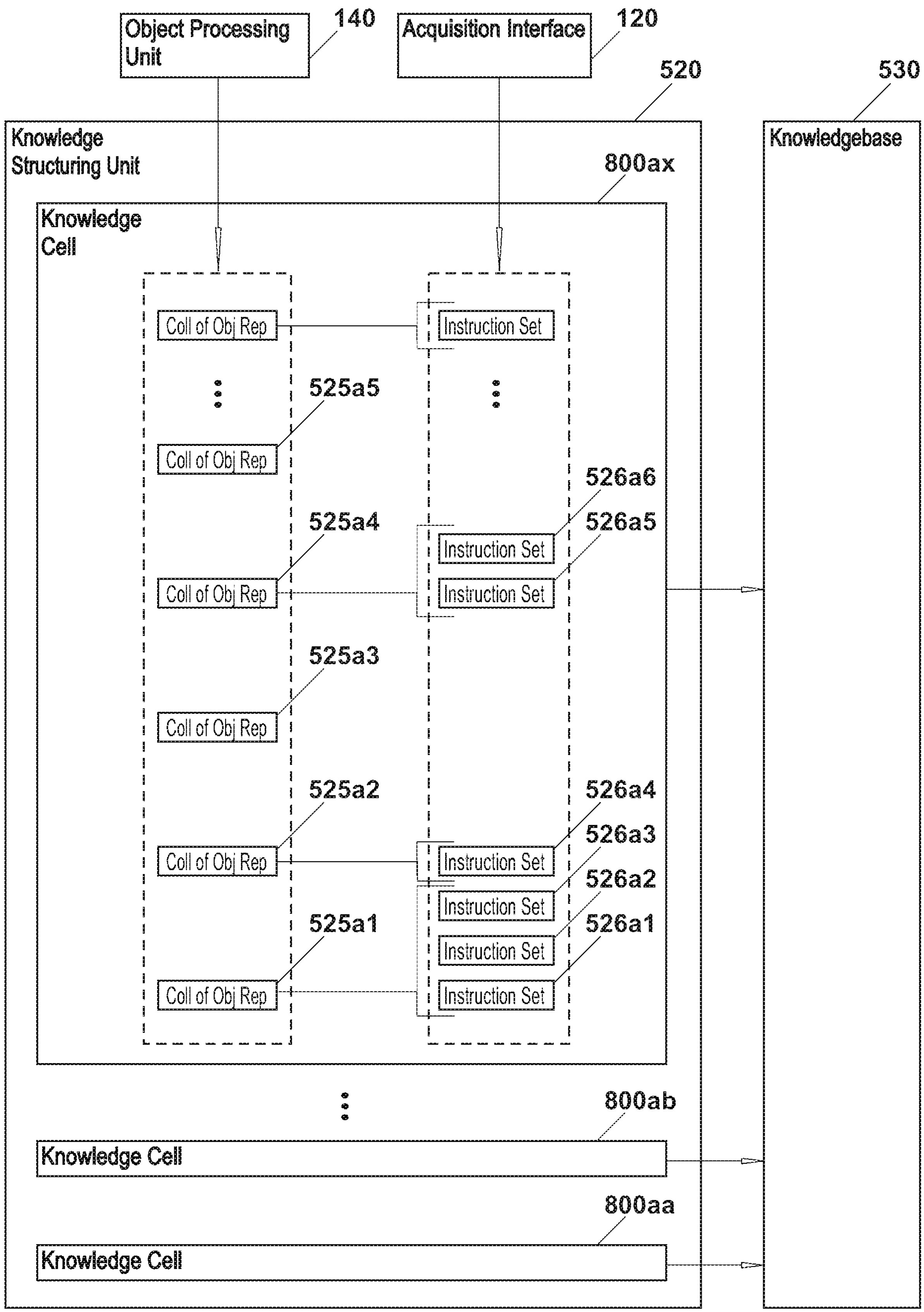
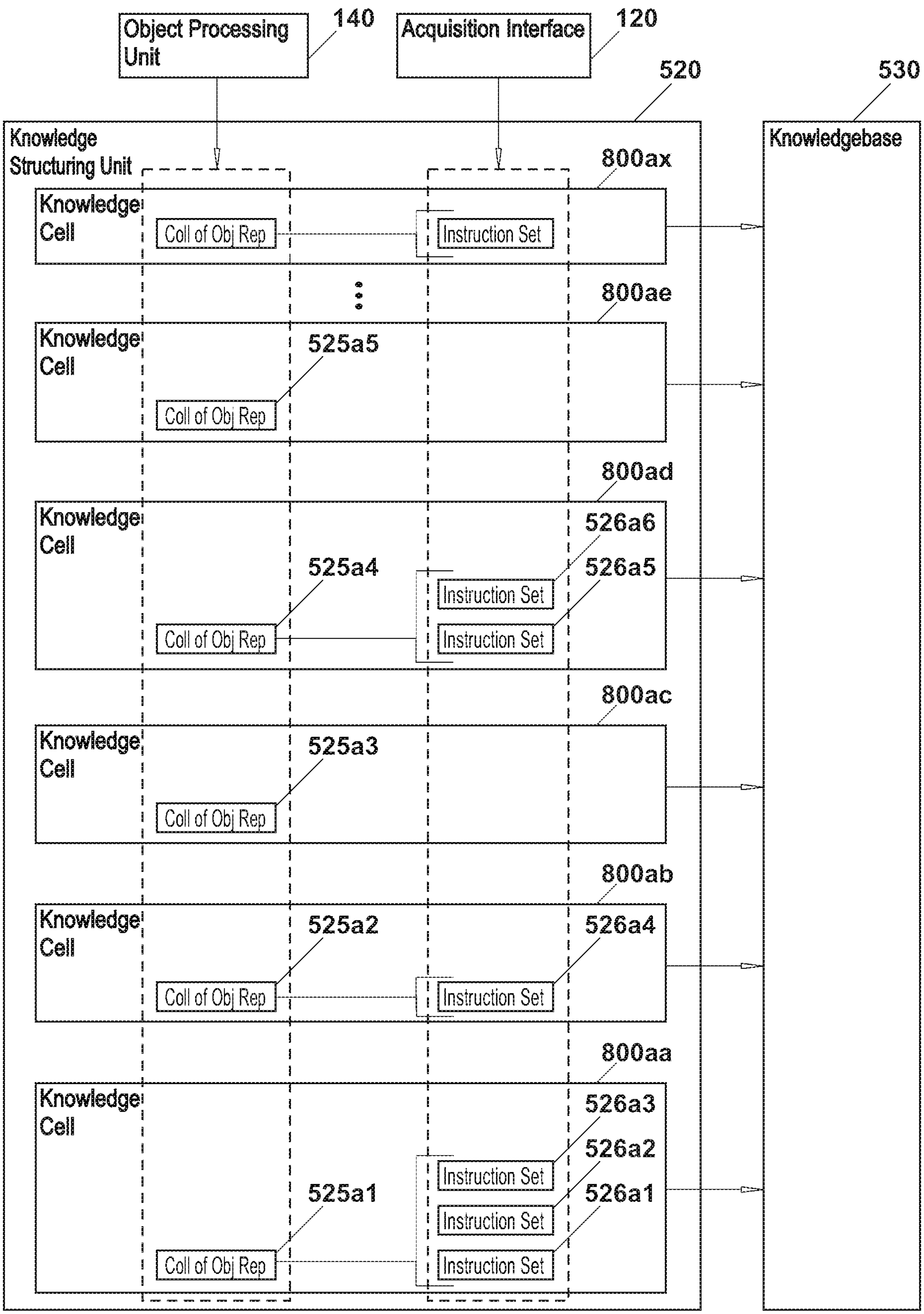

615a
615b
18
11
Processor
605
Application Program
Avatar
Object
Object
Object
100
ACAAO Unit
140
120
130
Object Processing Unit
Acquisition Interface
Modification Interface
110
Artificial Intelligence Unit
530
520
Knowledgebase
Knowledge Structuring Unit
540
Decision-making Unit
550
Confirmation Unit

FIG. 11



Object Processing Unit
140
Acquisition Interface
120
520
530
Knowledge Structuring Unit
Knowledgebase
800ax
Knowledge Cell
Coll of Obj Rep
Instruction Set
525a5
Coll of Obj Rep
526a6
525a4
526a5
Instruction Set
Coll of Obj Rep
Instruction Set
525a3
Coll of Obj Rep
525a2
526a4
526a3
Coll of Obj Rep
Instruction Set
526a2
Instruction Set
525a1
526a1
Instruction Set
Coll of Obj Rep
Instruction Set
800ab
Knowledge Cell
800aa
Knowledge Cell

FIG. 12



Object Processing Unit
140
Acquisition Interface
120
520
530
Knowledge Structuring Unit
Knowledgebase
800ax
Knowledge Cell
Coll of Obj Rep
Instruction Set
800ae
Knowledge Cell
525a5
Coll of Obj Rep
800ad
Knowledge Cell
526a6
525a4
526a5
Instruction Set
Coll of Obj Rep
Instruction Set
800ac
Knowledge Cell
525a3
Coll of Obj Rep
800ab
Knowledge Cell
525a2
526a4
Coll of Obj Rep
Instruction Set
800aa
Knowledge Cell
526a3
526a2
525a1
Instruction Set
526a1
Instruction Set
Coll of Obj Rep
Instruction Set

FIG. 13

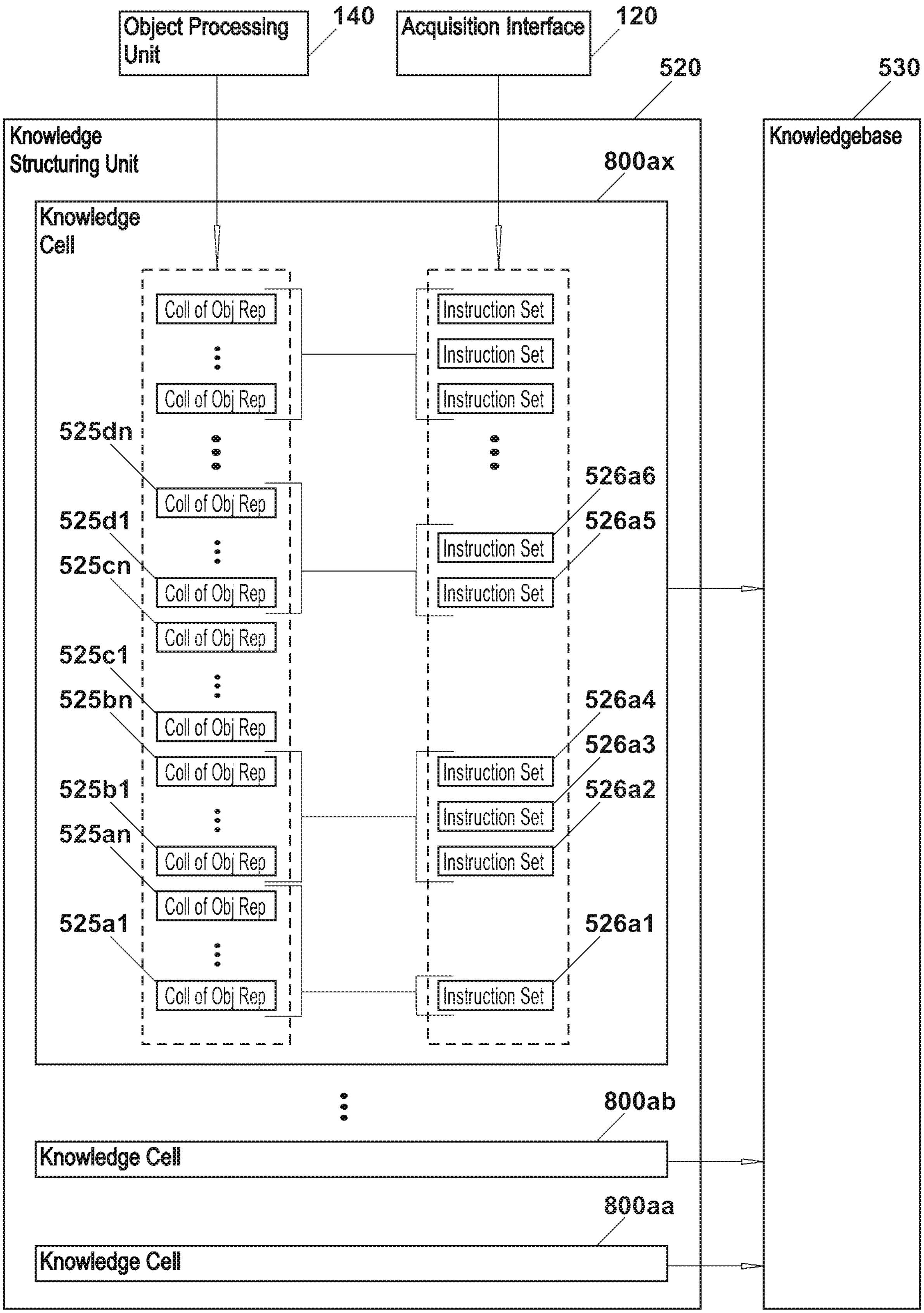
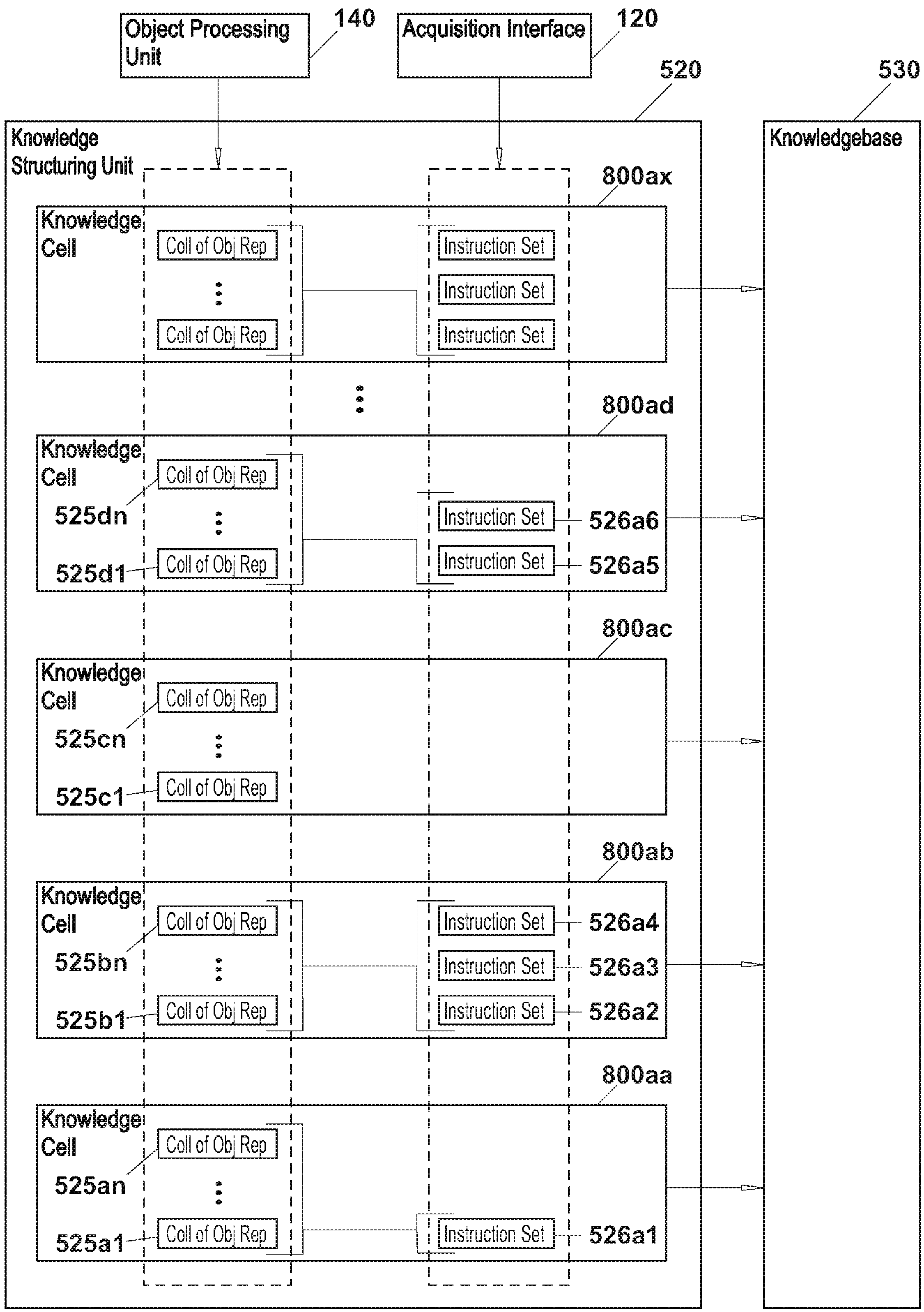

Object Processing Unit
140
Acquisition Interface
120
520
530
Knowledge Structuring Unit
800ax
Knowledgebase
Knowledge Cell
Coll of Obj Rep
Instruction Set
525dn
525d1
525cn
525c1
525bn
525b1
525an
525a1
526a6
526a5
526a4
526a3
526a2
526a1
800ab
Knowledge Cell
800aa
Knowledge Cell

FIG. 14



Object Processing Unit
140
Acquisition Interface
120
520
530
Knowledge Structuring Unit
Knowledgebase
800ax
Knowledge Cell
Coll of Obj Rep
Coll of Obj Rep
Instruction Set
Instruction Set
Instruction Set
800ad
Knowledge Cell
525dn
525d1
Coll of Obj Rep
Coll of Obj Rep
Instruction Set
Instruction Set
526a6
526a5
800ac
Knowledge Cell
525cn
525c1
Coll of Obj Rep
Coll of Obj Rep
800ab
Knowledge Cell
525bn
525b1
Coll of Obj Rep
Coll of Obj Rep
Instruction Set
Instruction Set
Instruction Set
526a4
526a3
526a2
800aa
Knowledge Cell
525an
525a1
Coll of Obj Rep
Coll of Obj Rep
Instruction Set
526a1

FIG. 15

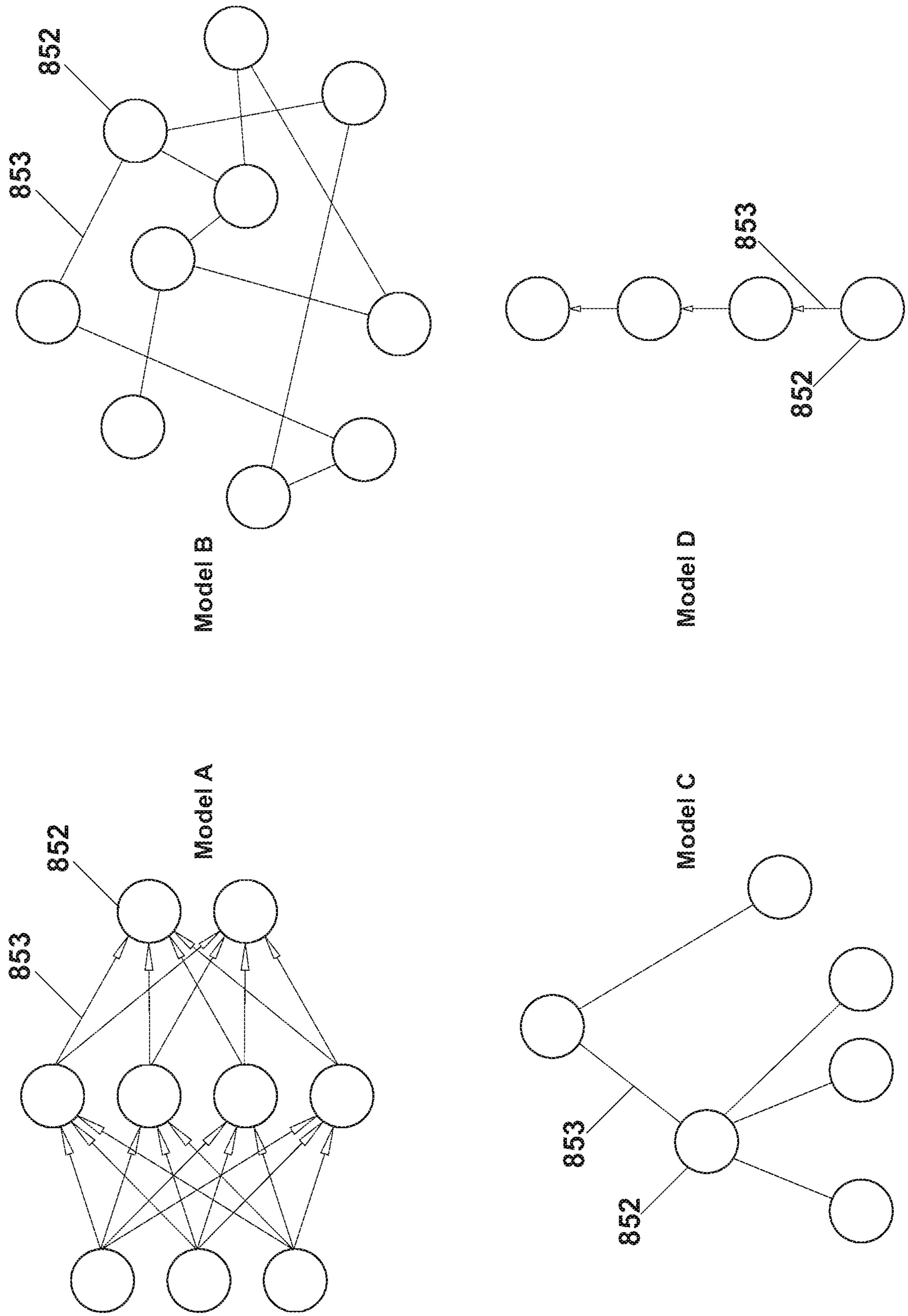


FIG. 16

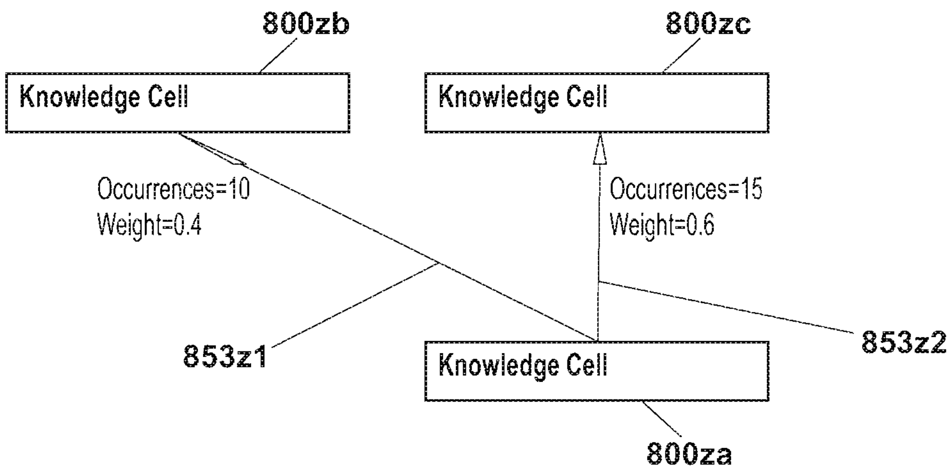


FIG. 17A

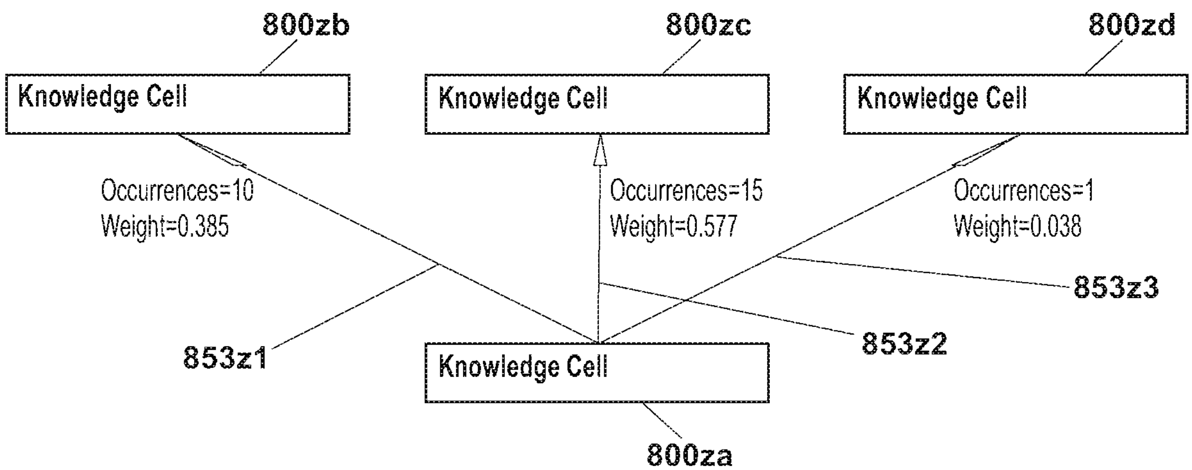


FIG. 17B

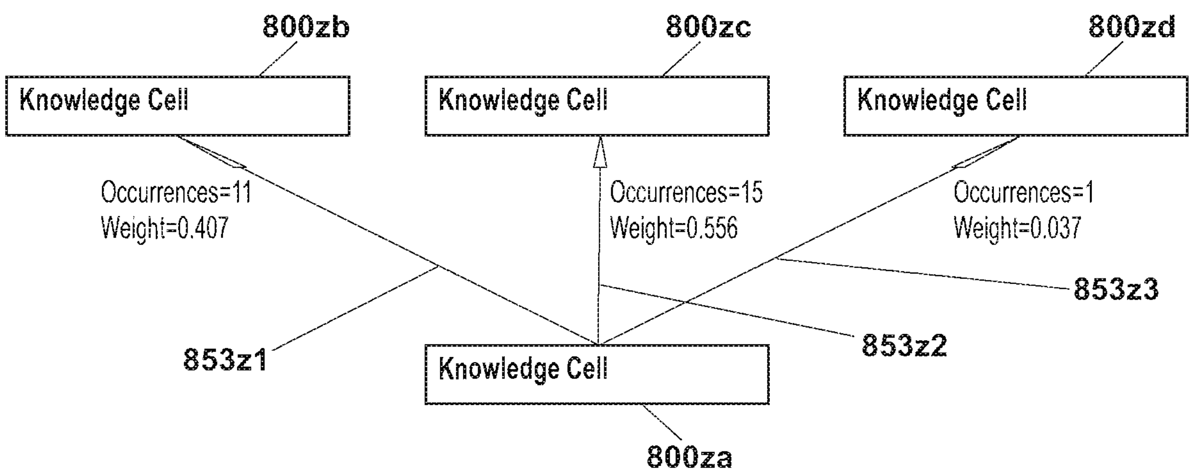


FIG. 17C

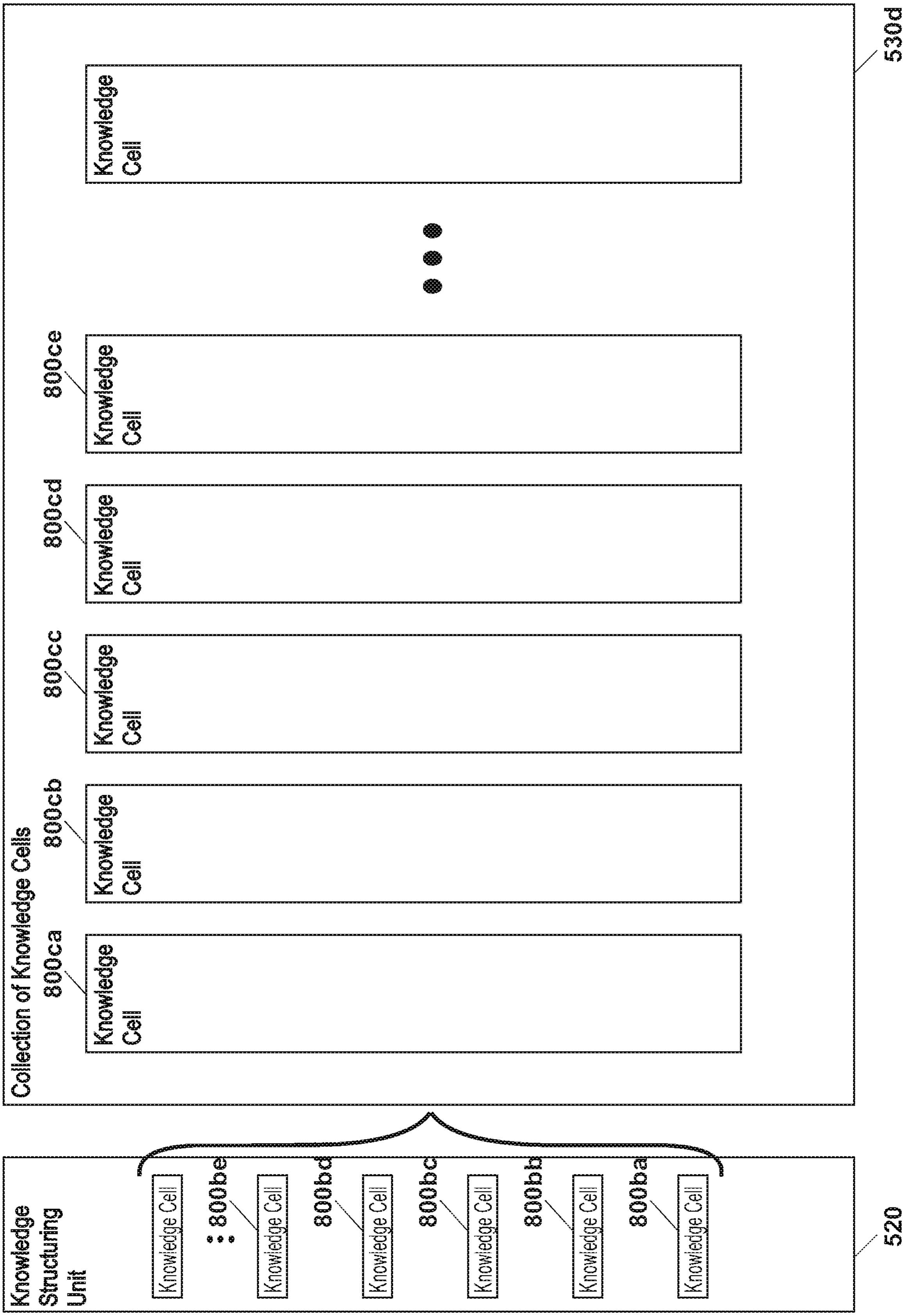

Knowledge Structuring Unit
Knowledge Cell
Knowledge Cell
Knowledge Cell
Knowledge Cell
Knowledge Cell
Knowledge Cell
800ba
800bb
800bc
800bd
800be
520
Collection of Knowledge Cells
Knowledge Cell
Knowledge Cell
Knowledge Cell
Knowledge Cell
Knowledge Cell
Knowledge Cell
800ca
800cb
800cc
800cd
800ce
530d

FIG. 18

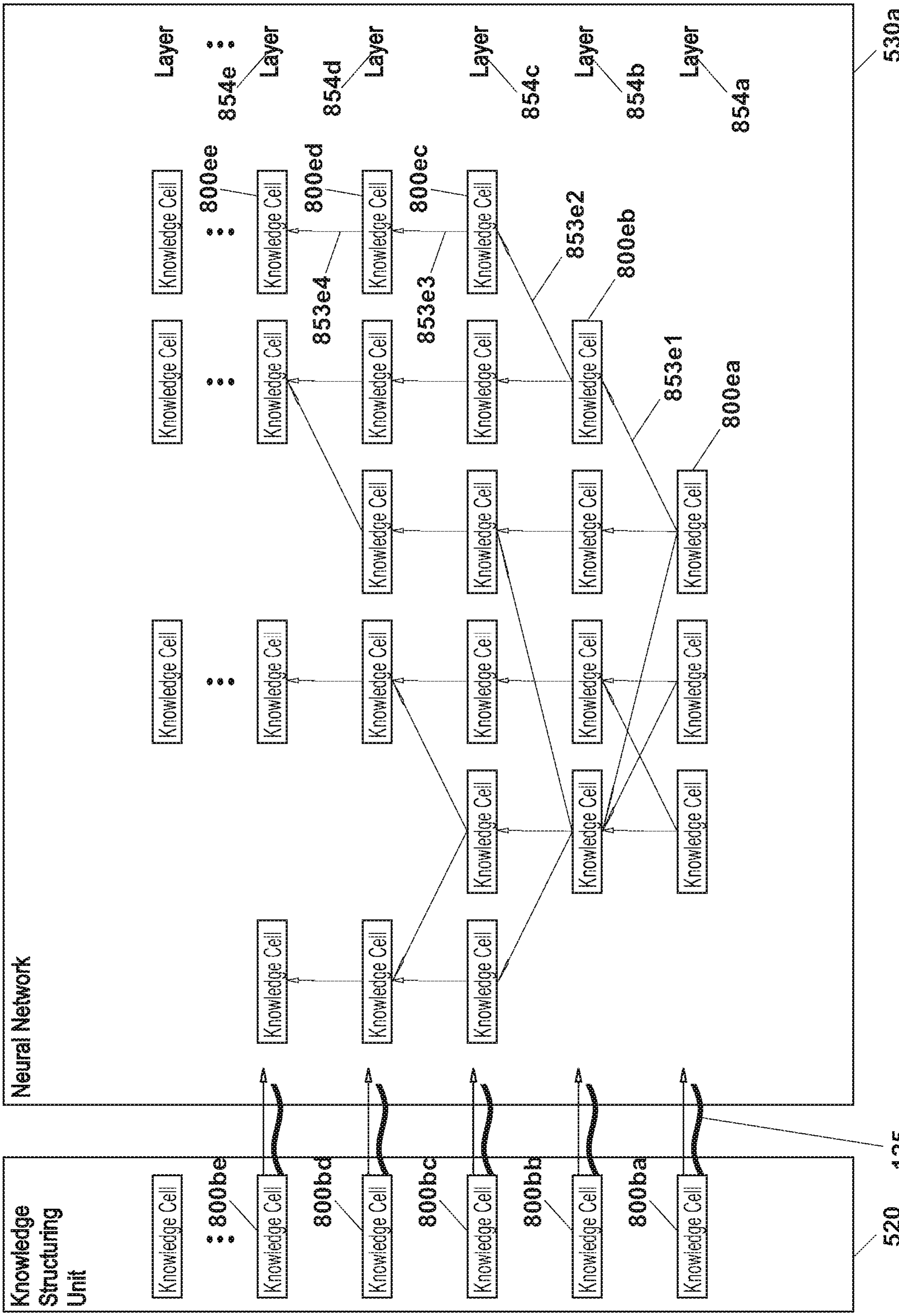
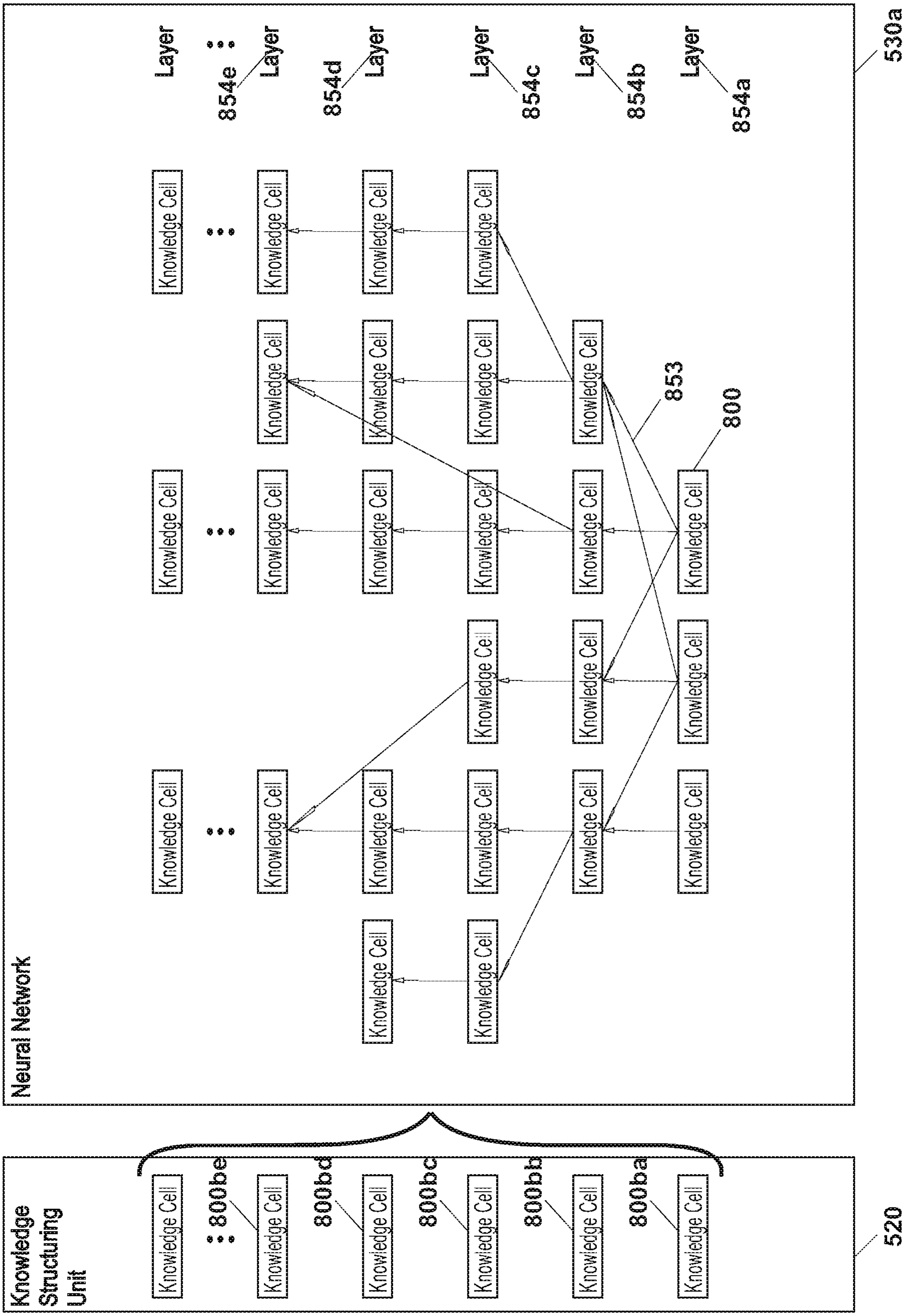

Knowledge Structuring Unit
Neural Network
Knowledge Cell
Layer
530a
520
125
800ba
800bb
800bc
800bd
800be
800ea
800eb
800ec
800ed
800ee
853e1
853e2
853e3
853e4
854a
854b
854c
854d
854e

FIG. 19



Neural Network
Layer
Layer
Layer
Layer
Layer
Layer
854e
854d
854c
854b
854a
530a
Knowledge Cell
853
800
Knowledge Structuring Unit
800be
800bd
800bc
800bb
800ba
520

FIG. 20

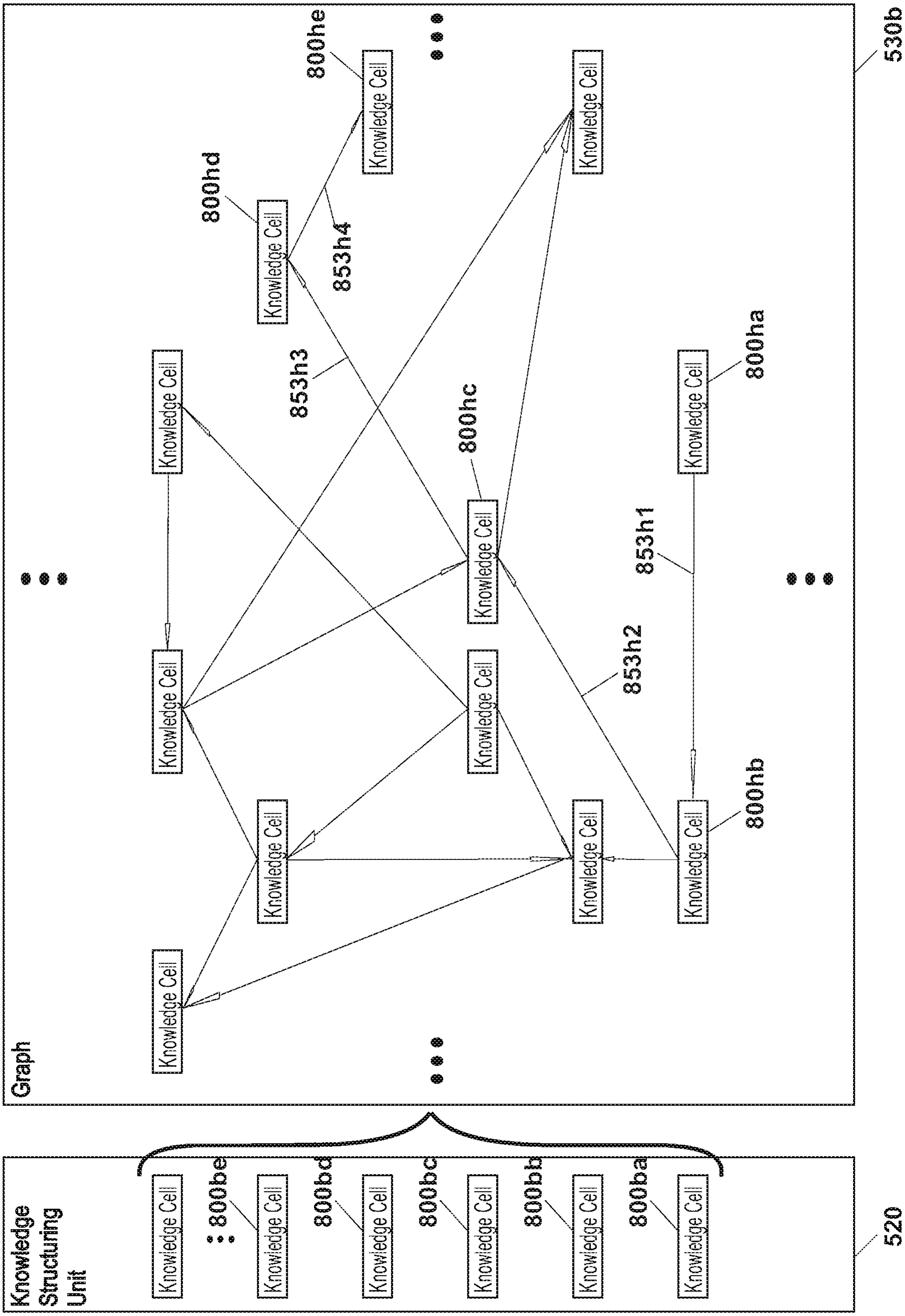

Graph
Knowledge Cell
800ha
800hb
800hc
800hd
800he
853h1
853h2
853h3
853h4
530b
Knowledge Structuring Unit
800ba
800bb
800bc
800bd
800be
520

FIG. 21

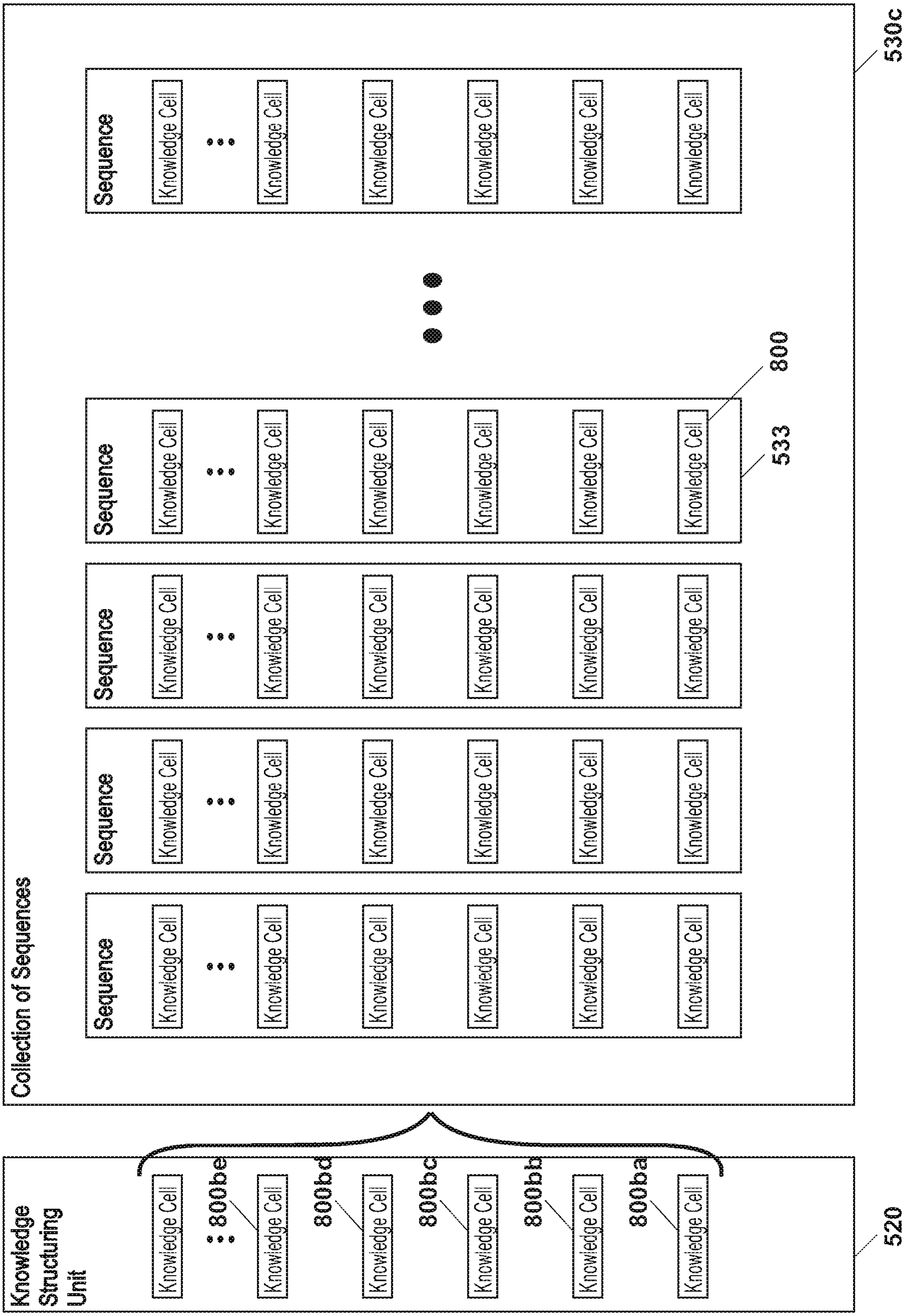

Knowledge Structuring Unit
Knowledge Cell
800be
800bd
800bc
800bb
800ba
520
Collection of Sequences
Sequence
Knowledge Cell
533
800
530c

FIG. 22

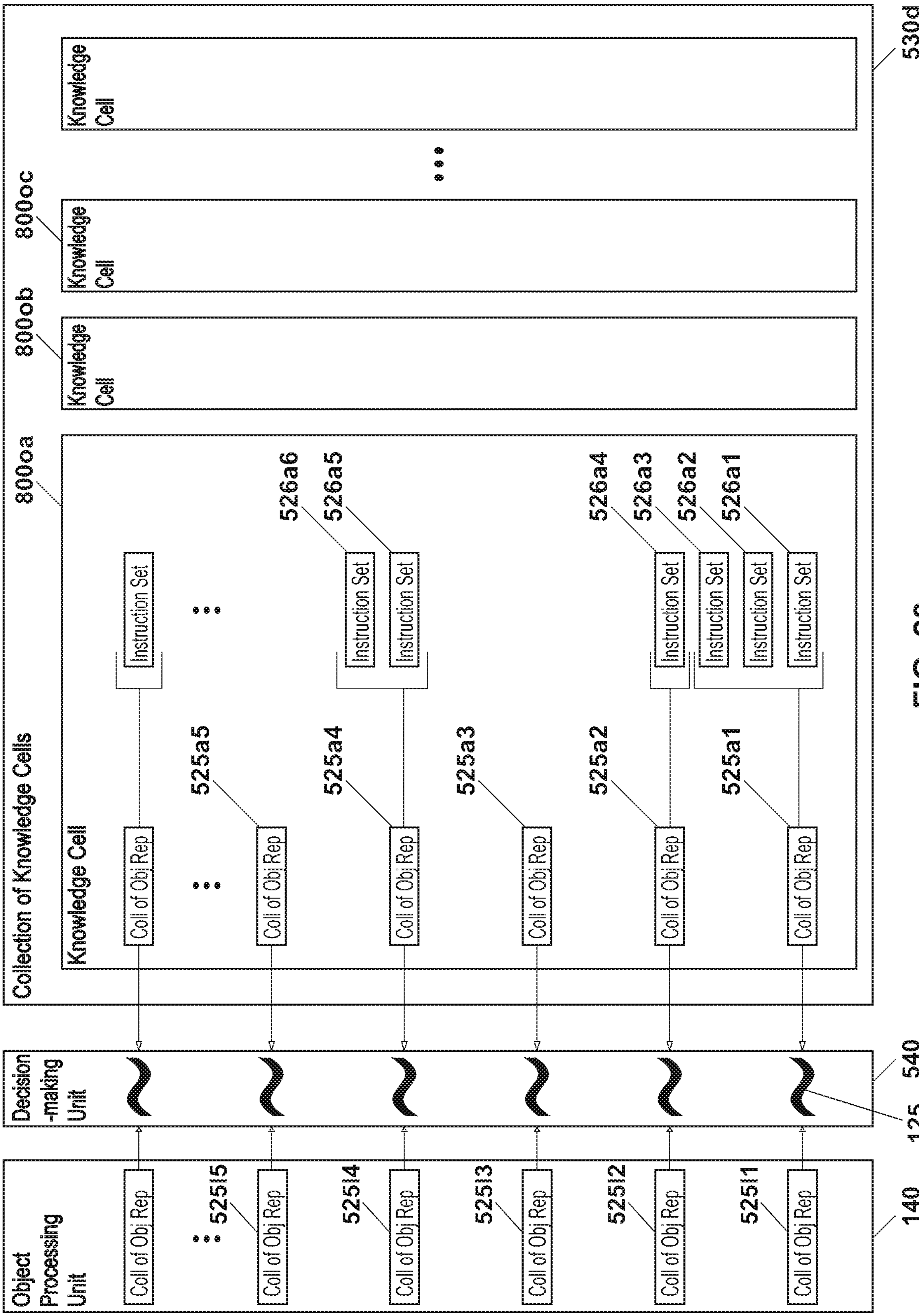
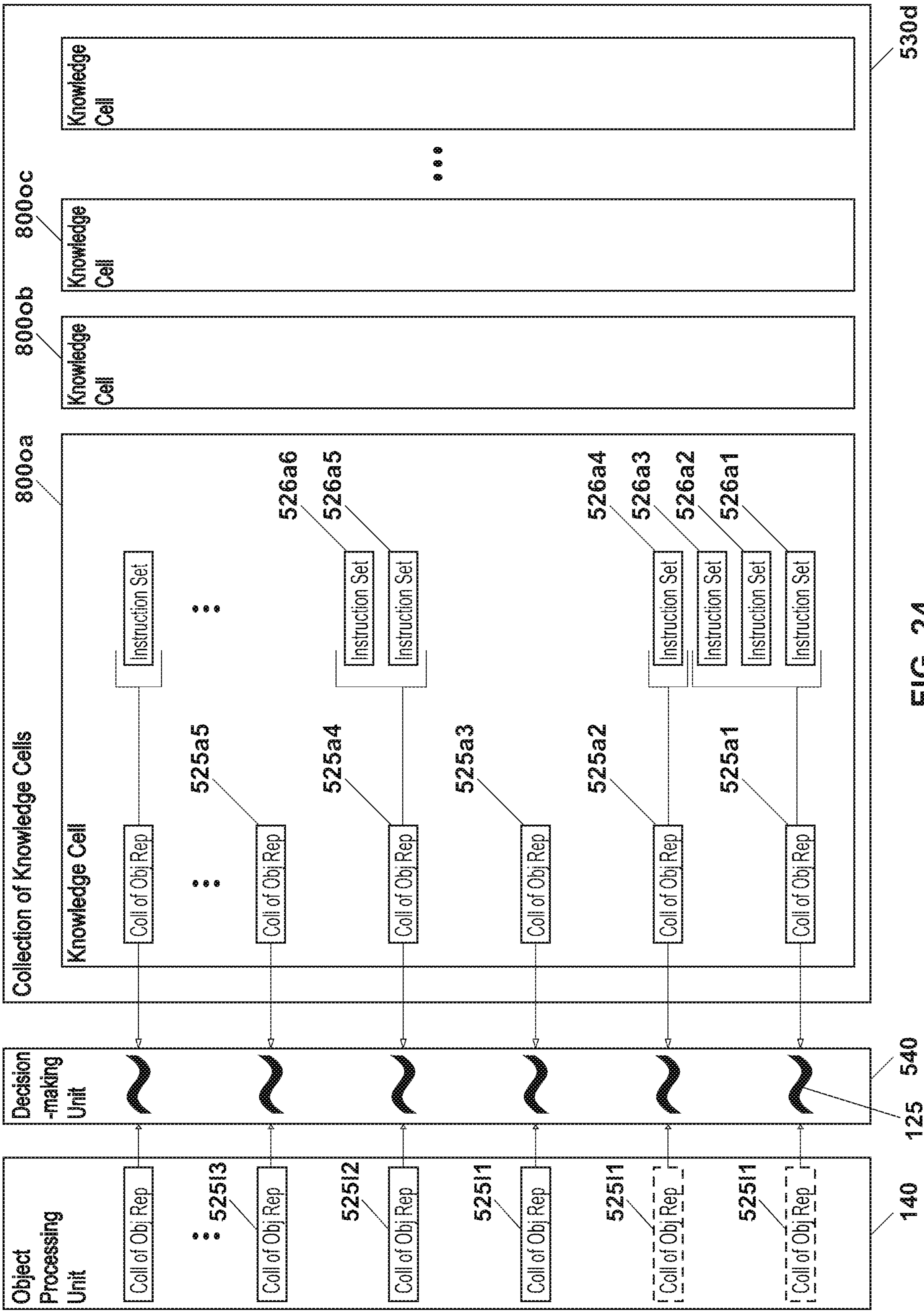

Object Processing Unit
Coll of Obj Rep
525l5
525l4
525l3
525l2
525l1
140
Decision -making Unit
125
540
Collection of Knowledge Cells
Knowledge Cell
Coll of Obj Rep
Instruction Set
525a5
525a4
525a3
525a2
525a1
526a6
526a5
526a4
526a3
526a2
526a1
800oa
800ob
800oc
Knowledge Cell
530d

FIG. 23



Object Processing Unit
Coll of Obj Rep
525l3
525l2
525l1
140
Decision -making Unit
125
540
Collection of Knowledge Cells
Knowledge Cell
525a5
525a4
525a3
525a2
525a1
Instruction Set
526a6
526a5
526a4
526a3
526a2
526a1
800oa
800ob
800oc
530d

FIG. 24

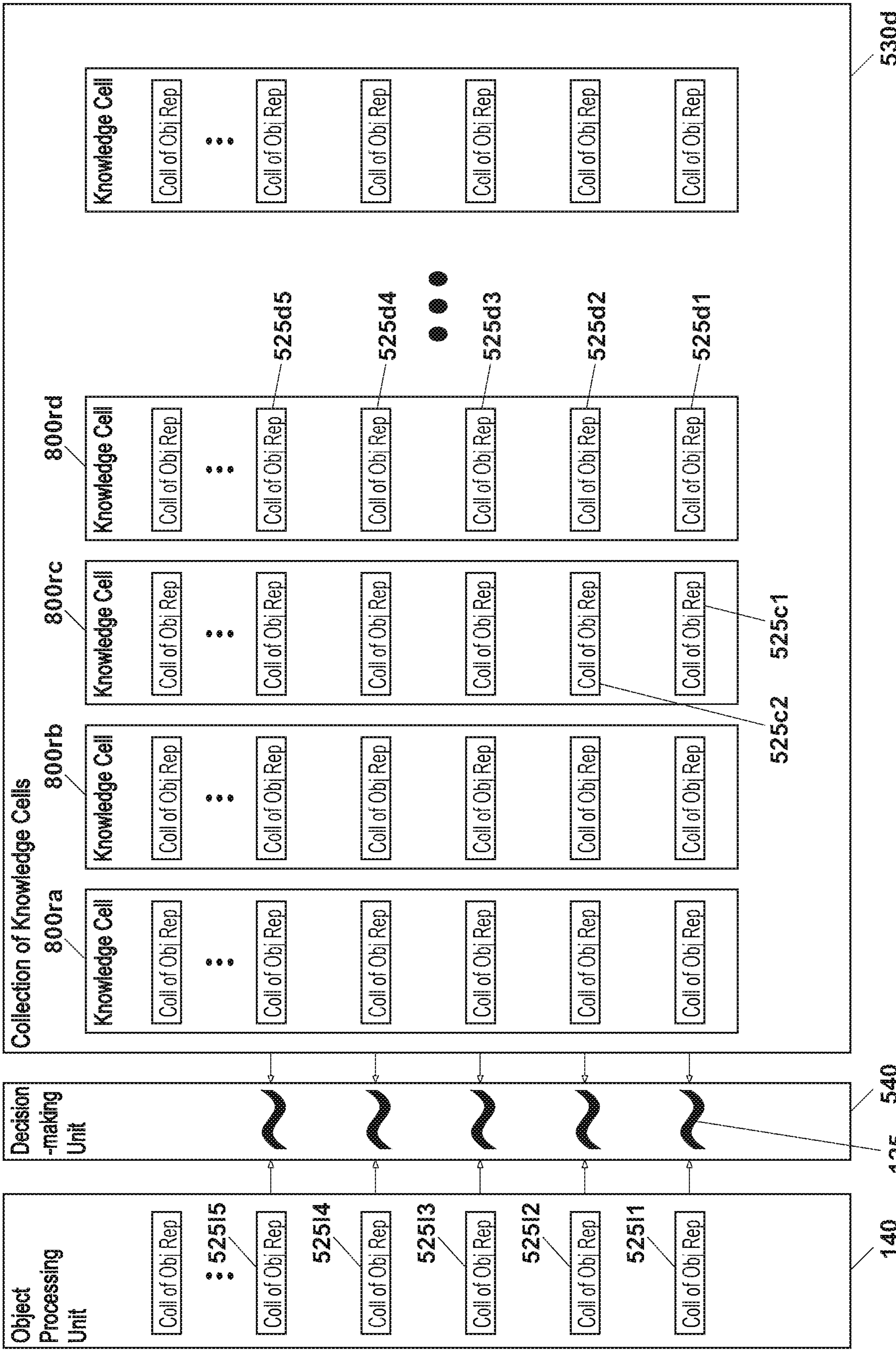

Object Processing Unit
Coll of Obj Rep
525l5
525l4
525l3
525l2
525l1
140
Decision -making Unit
125
540
Collection of Knowledge Cells
800ra
800rb
800rc
800rd
Knowledge Cell
525c1
525c2
525d1
525d2
525d3
525d4
525d5
530d

FIG. 25

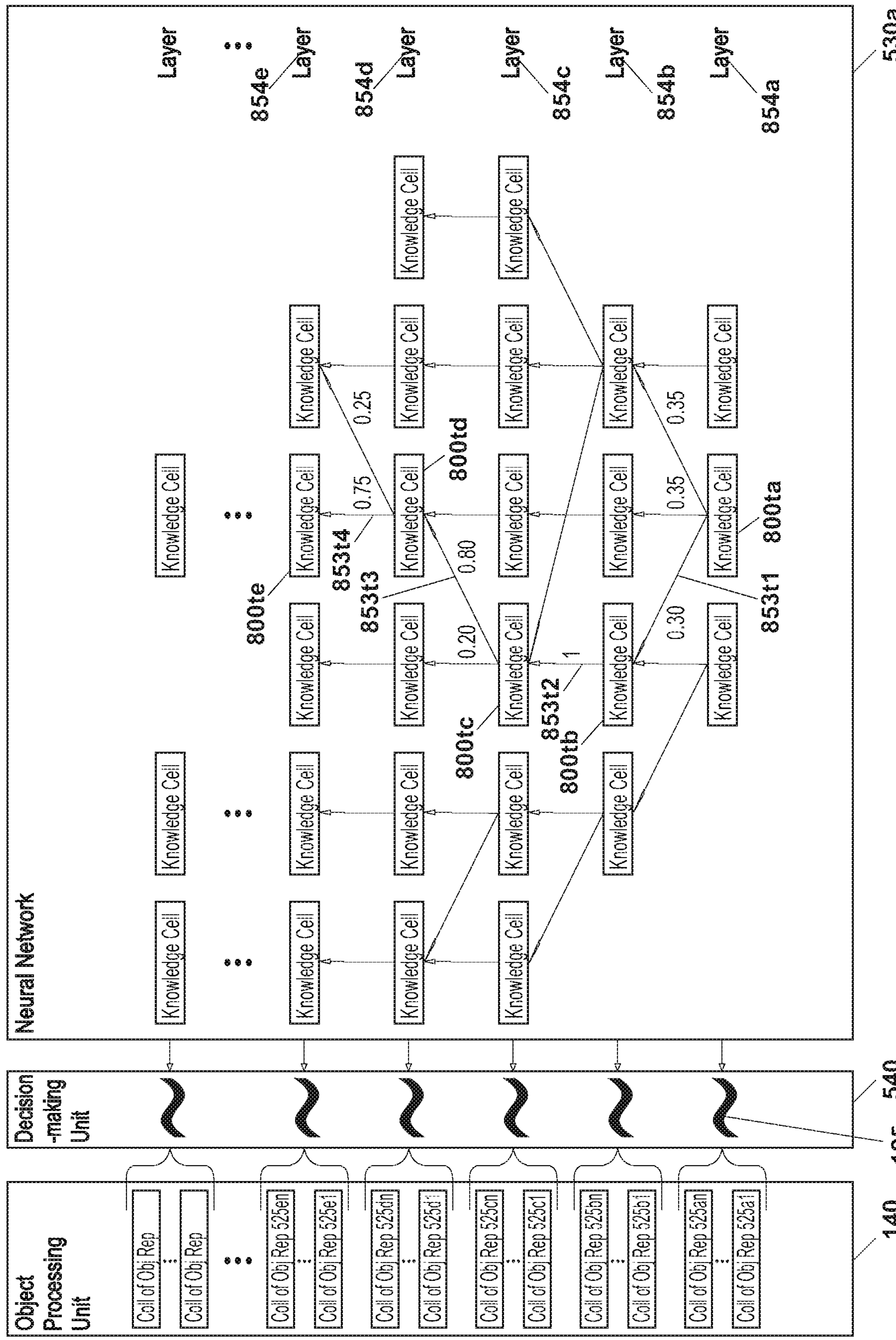
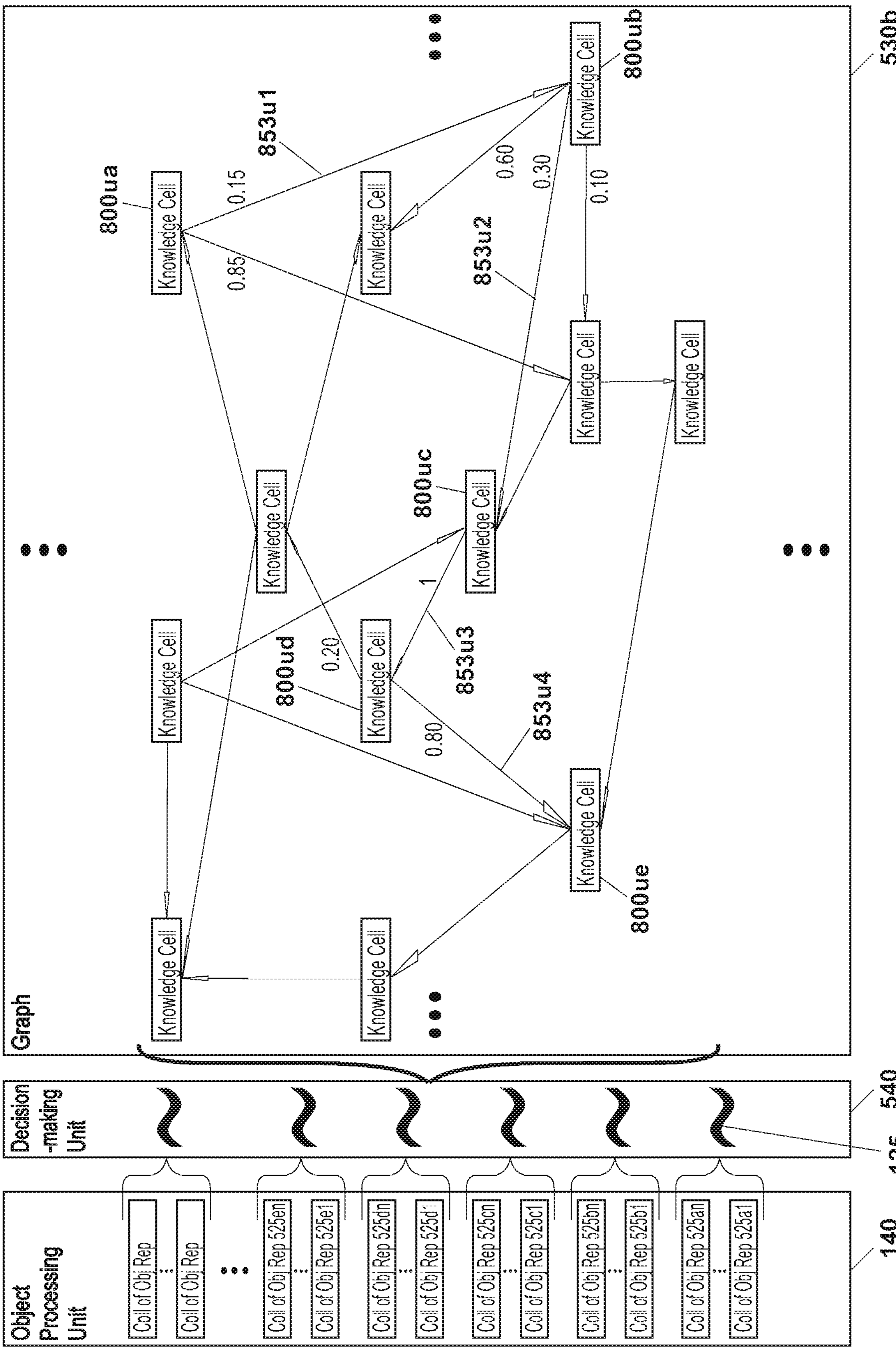

Object Processing Unit
Coll of Obj Rep
Coll of Obj Rep 525en
Coll of Obj Rep 525e1
Coll of Obj Rep 525dn
Coll of Obj Rep 525d1
Coll of Obj Rep 525cn
Coll of Obj Rep 525c1
Coll of Obj Rep 525bn
Coll of Obj Rep 525b1
Coll of Obj Rep 525an
Coll of Obj Rep 525a1
140
Decision -making Unit
125
540
Neural Network
Knowledge Cell
800te
800td
800tc
800tb
800ta
853t1
853t2
853t3
853t4
0.75
0.25
0.80
0.20
1
0.35
0.35
0.30
Layer
854e
854d
854c
854b
854a
530a

FIG. 26



Graph
Knowledge Cell
800ua
800ub
800uc
800ud
800ue
853u1
853u2
853u3
853u4
0.15
0.85
0.60
0.30
0.10
0.20
0.80
1
530b
Decision -making Unit
540
125
Object Processing Unit
Coll of Obj Rep
Coll of Obj Rep 525en
Coll of Obj Rep 525e1
Coll of Obj Rep 525dn
Coll of Obj Rep 525d1
Coll of Obj Rep 525cn
Coll of Obj Rep 525c1
Coll of Obj Rep 525bn
Coll of Obj Rep 525b1
Coll of Obj Rep 525an
Coll of Obj Rep 525a1
140

FIG. 27

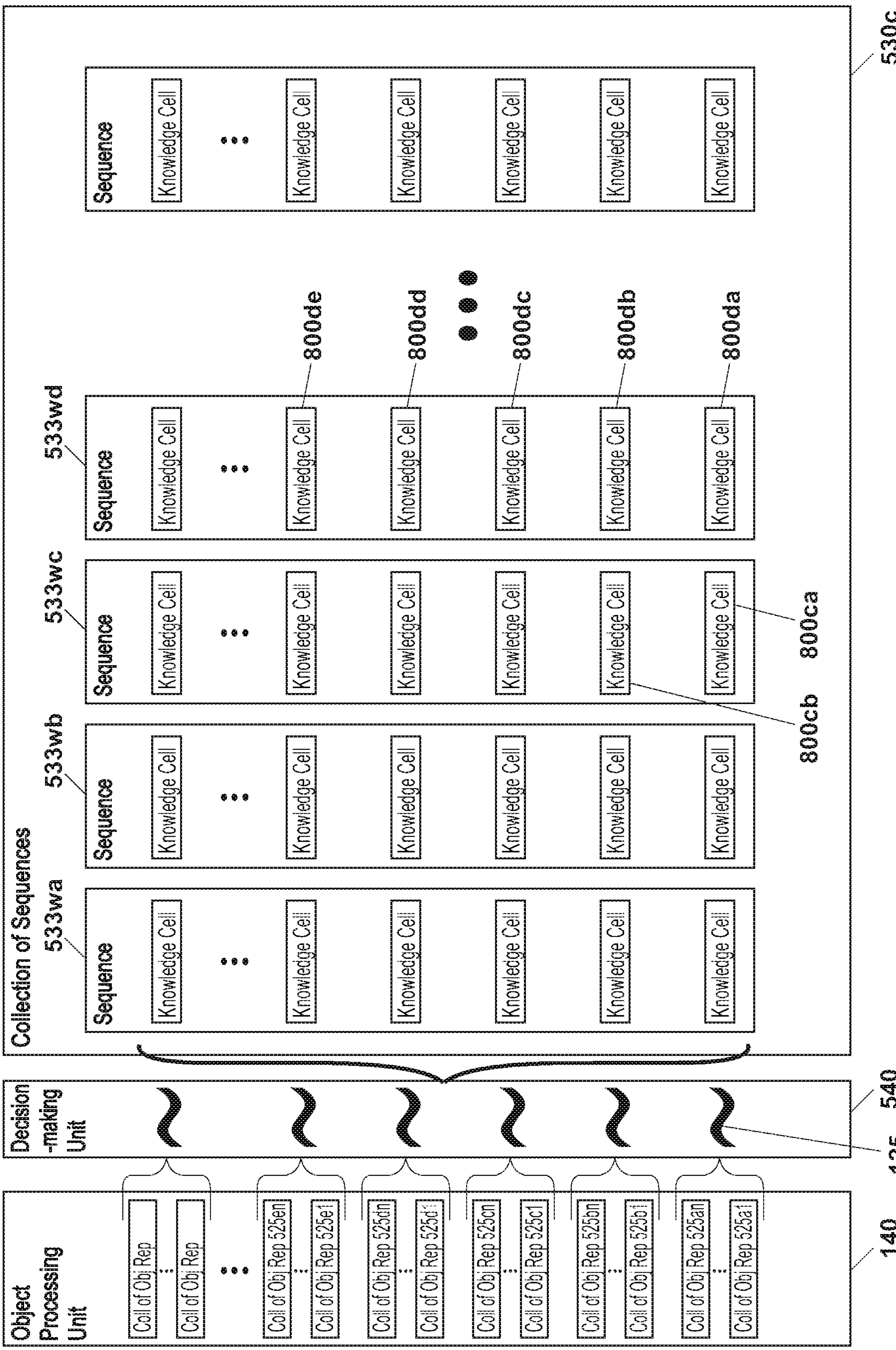

Object Processing Unit
Coll of Obj Rep
Coll of Obj Rep
Coll of Obj Rep 525en
Coll of Obj Rep 525e1
Coll of Obj Rep 525dn
Coll of Obj Rep 525d1
Coll of Obj Rep 525cn
Coll of Obj Rep 525c1
Coll of Obj Rep 525bn
Coll of Obj Rep 525b1
Coll of Obj Rep 525an
Coll of Obj Rep 525a1
140
Decision -making Unit
540
125
Collection of Sequences
533wa
533wb
533wc
533wd
Sequence
Knowledge Cell
800ca
800cb
800da
800db
800dc
800dd
800de
530c

FIG. 28

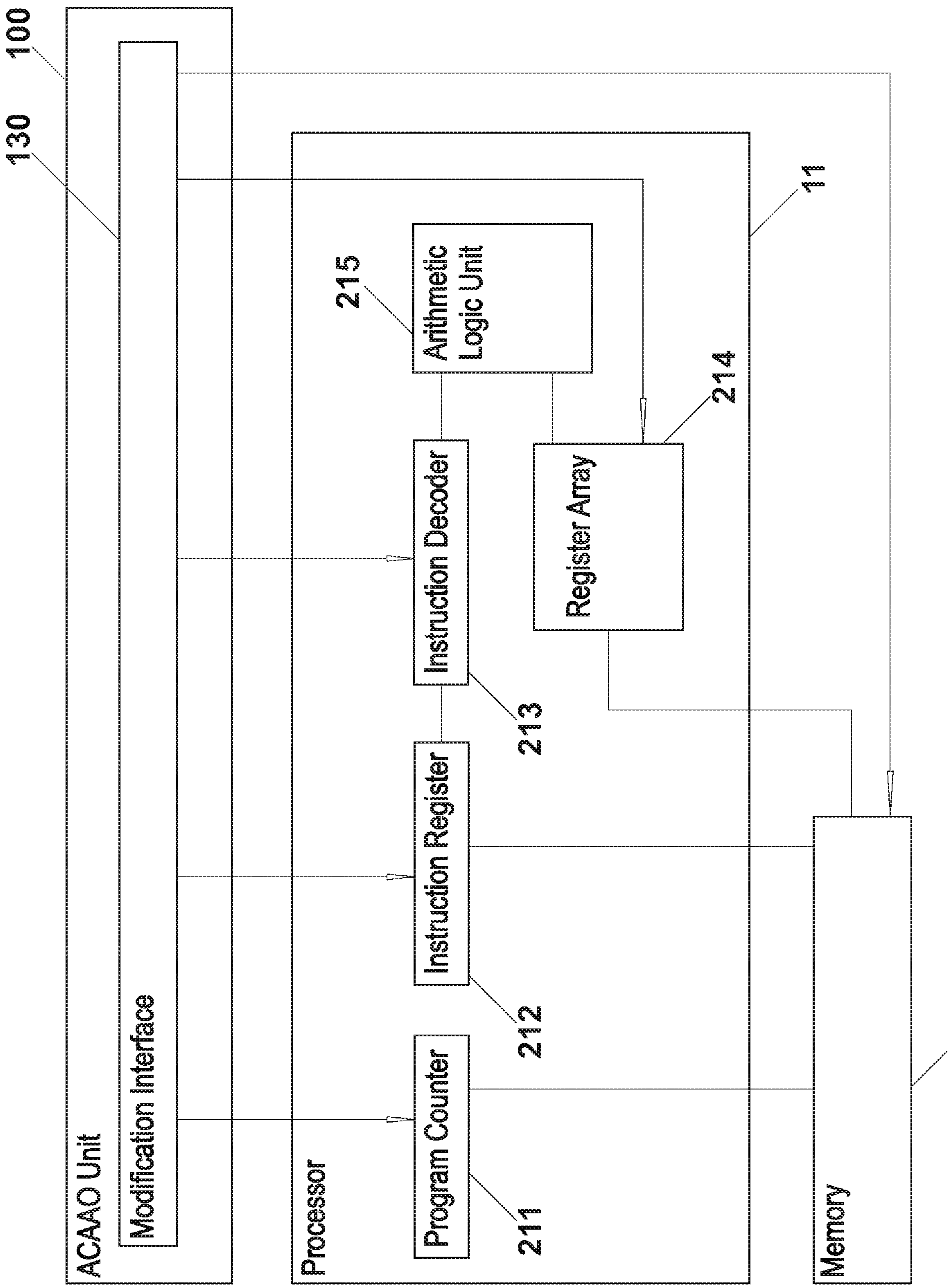

100
130
ACAAO Unit
Modification Interface
Processor
Program Counter
Instruction Register
Instruction Decoder
Arithmetic Logic Unit
Register Array
Memory
211
212
213
214
215
11
12

FIG. 29

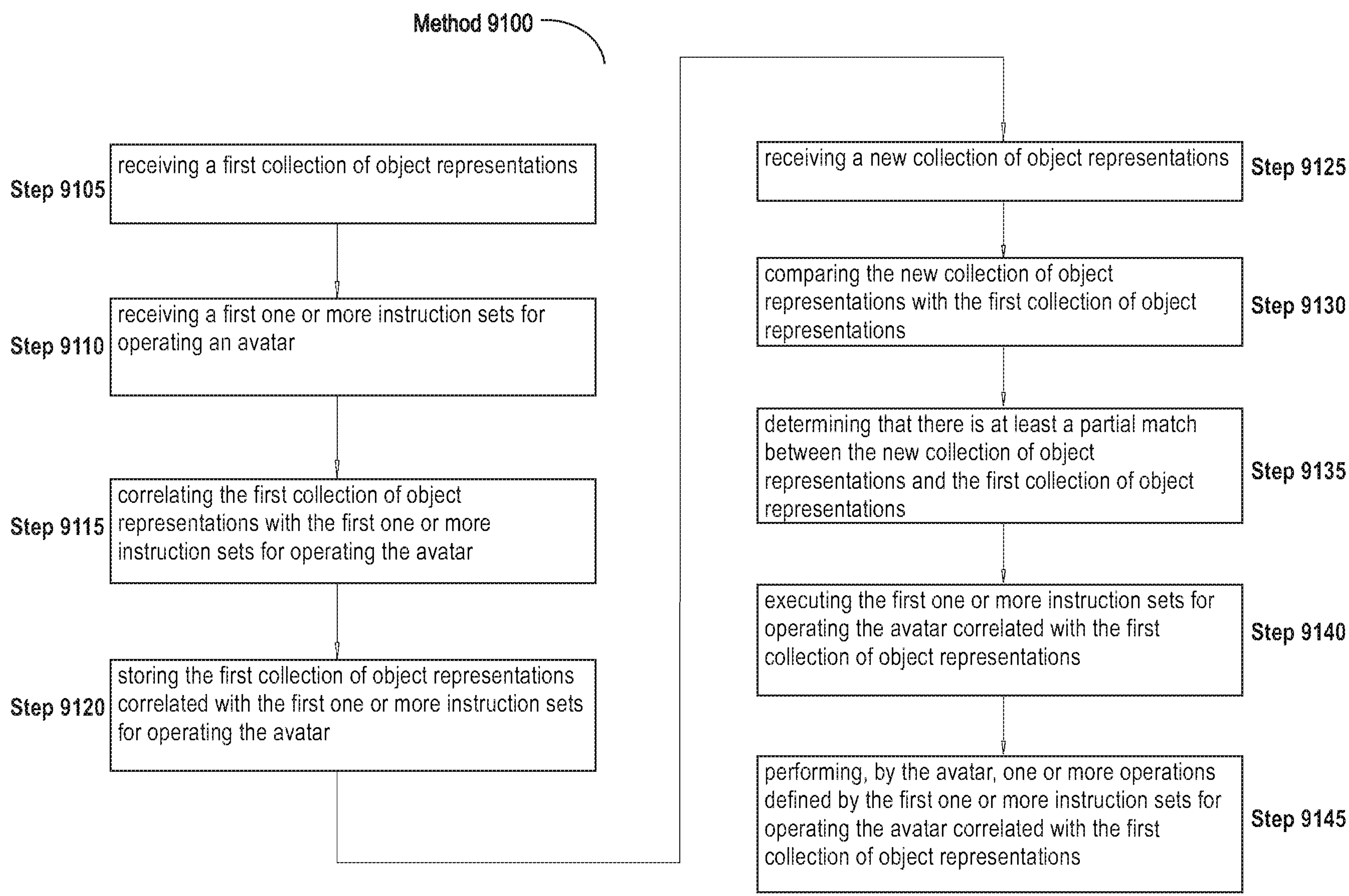

Method 9100
Step 9105
receiving a first collection of object representations
Step 9110
receiving a first one or more instruction sets for operating an avatar
Step 9115
correlating the first collection of object representations with the first one or more instruction sets for operating the avatar
Step 9120
storing the first collection of object representations correlated with the first one or more instruction sets for operating the avatar
Step 9125
receiving a new collection of object representations
Step 9130
comparing the new collection of object representations with the first collection of object representations
Step 9135
determining that there is at least a partial match between the new collection of object representations and the first collection of object representations
Step 9140
executing the first one or more instruction sets for operating the avatar correlated with the first collection of object representations
Step 9145
performing, by the avatar, one or more operations defined by the first one or more instruction sets for operating the avatar correlated with the first collection of object representations

FIG. 30

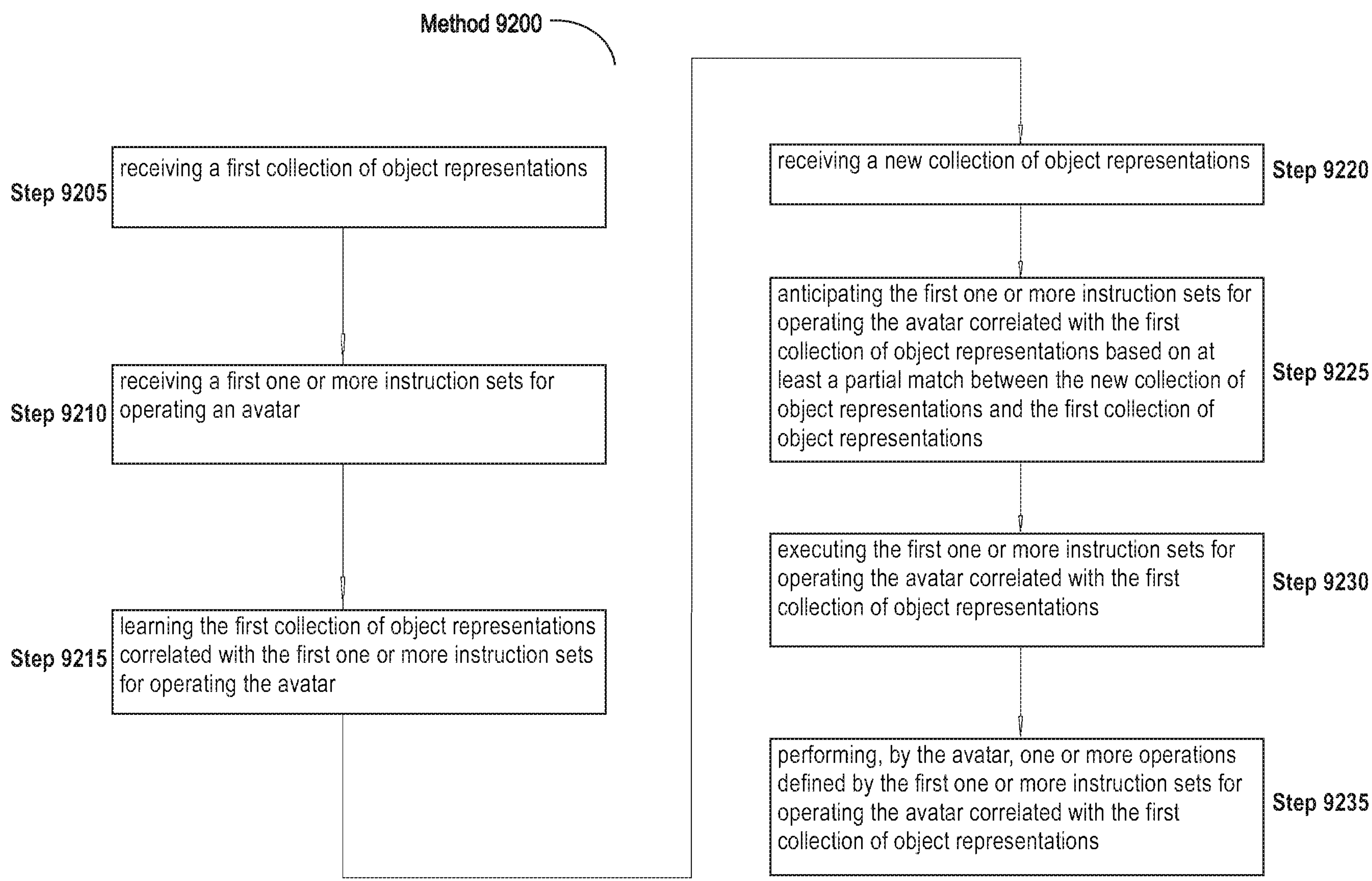

Method 9200
Step 9205
receiving a first collection of object representations
Step 9210
receiving a first one or more instruction sets for operating an avatar
Step 9215
learning the first collection of object representations correlated with the first one or more instruction sets for operating the avatar
Step 9220
receiving a new collection of object representations
Step 9225
anticipating the first one or more instruction sets for operating the avatar correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations
Step 9230
executing the first one or more instruction sets for operating the avatar correlated with the first collection of object representations
Step 9235
performing, by the avatar, one or more operations defined by the first one or more instruction sets for operating the avatar correlated with the first collection of object representations

FIG. 31

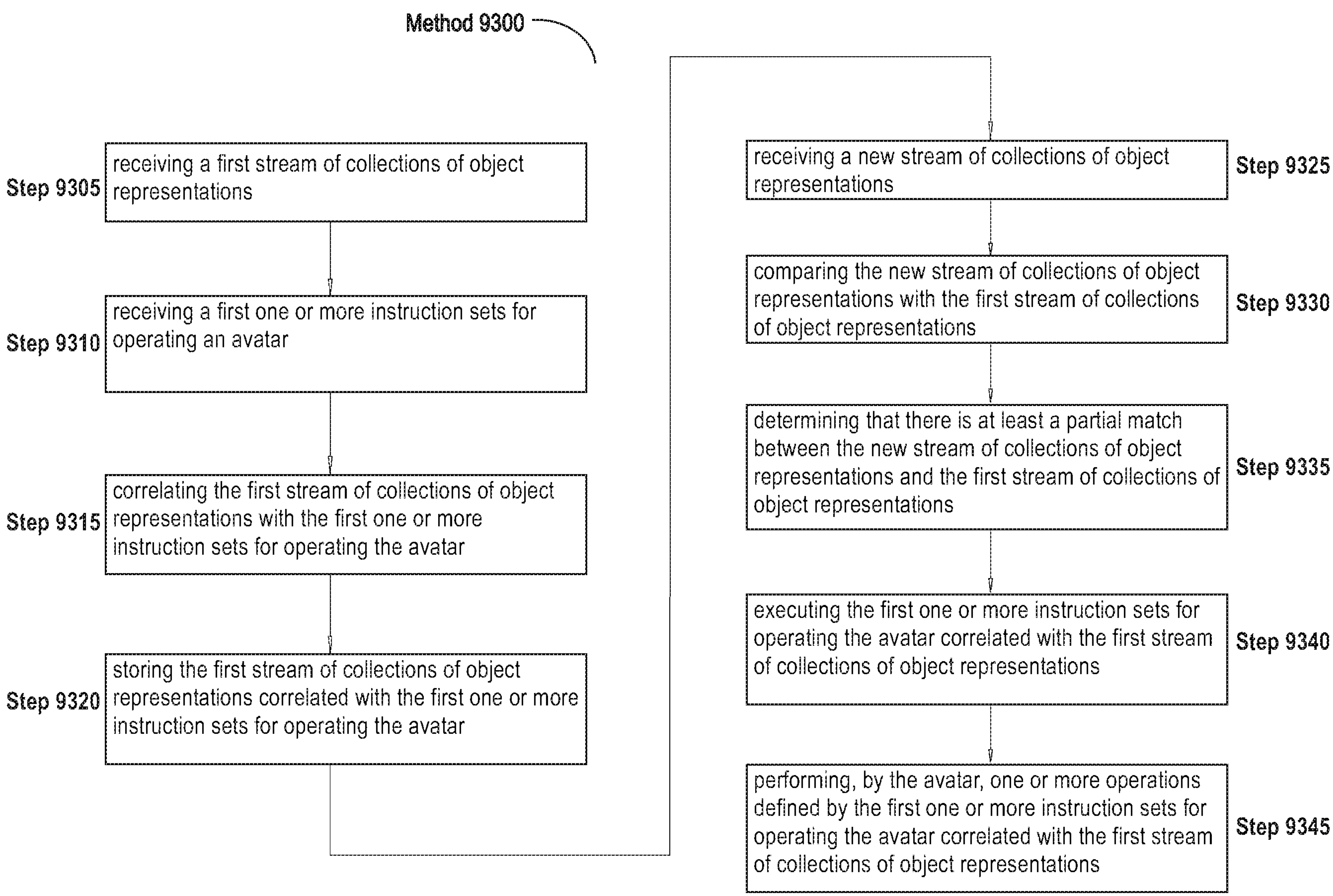

Method 9300
Step 9305
receiving a first stream of collections of object representations
Step 9310
receiving a first one or more instruction sets for operating an avatar
Step 9315
correlating the first stream of collections of object representations with the first one or more instruction sets for operating the avatar
Step 9320
storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar
Step 9325
receiving a new stream of collections of object representations
Step 9330
comparing the new stream of collections of object representations with the first stream of collections of object representations
Step 9335
determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations
Step 9340
executing the first one or more instruction sets for operating the avatar correlated with the first stream of collections of object representations
Step 9345
performing, by the avatar, one or more operations defined by the first one or more instruction sets for operating the avatar correlated with the first stream of collections of object representations

FIG. 32

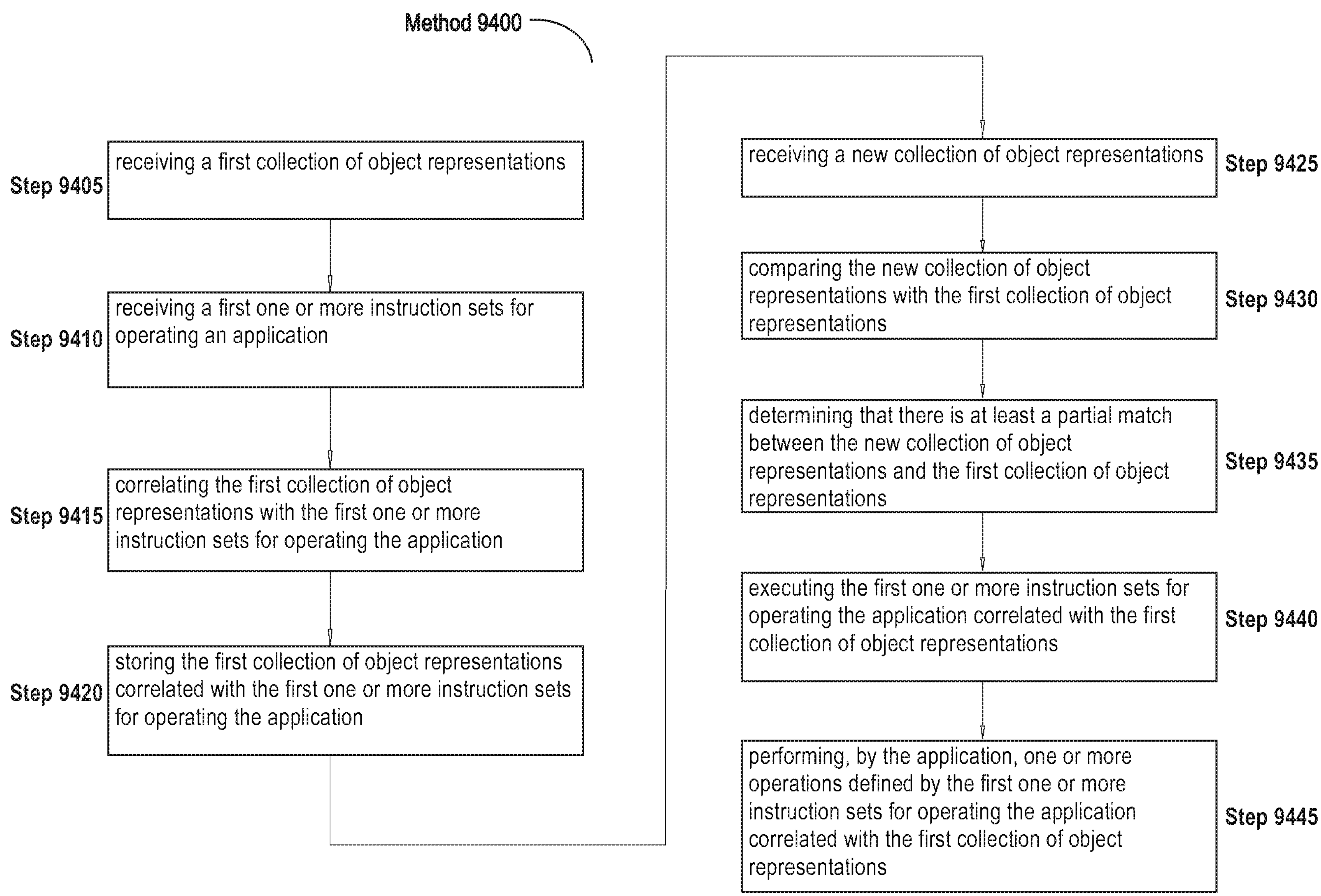
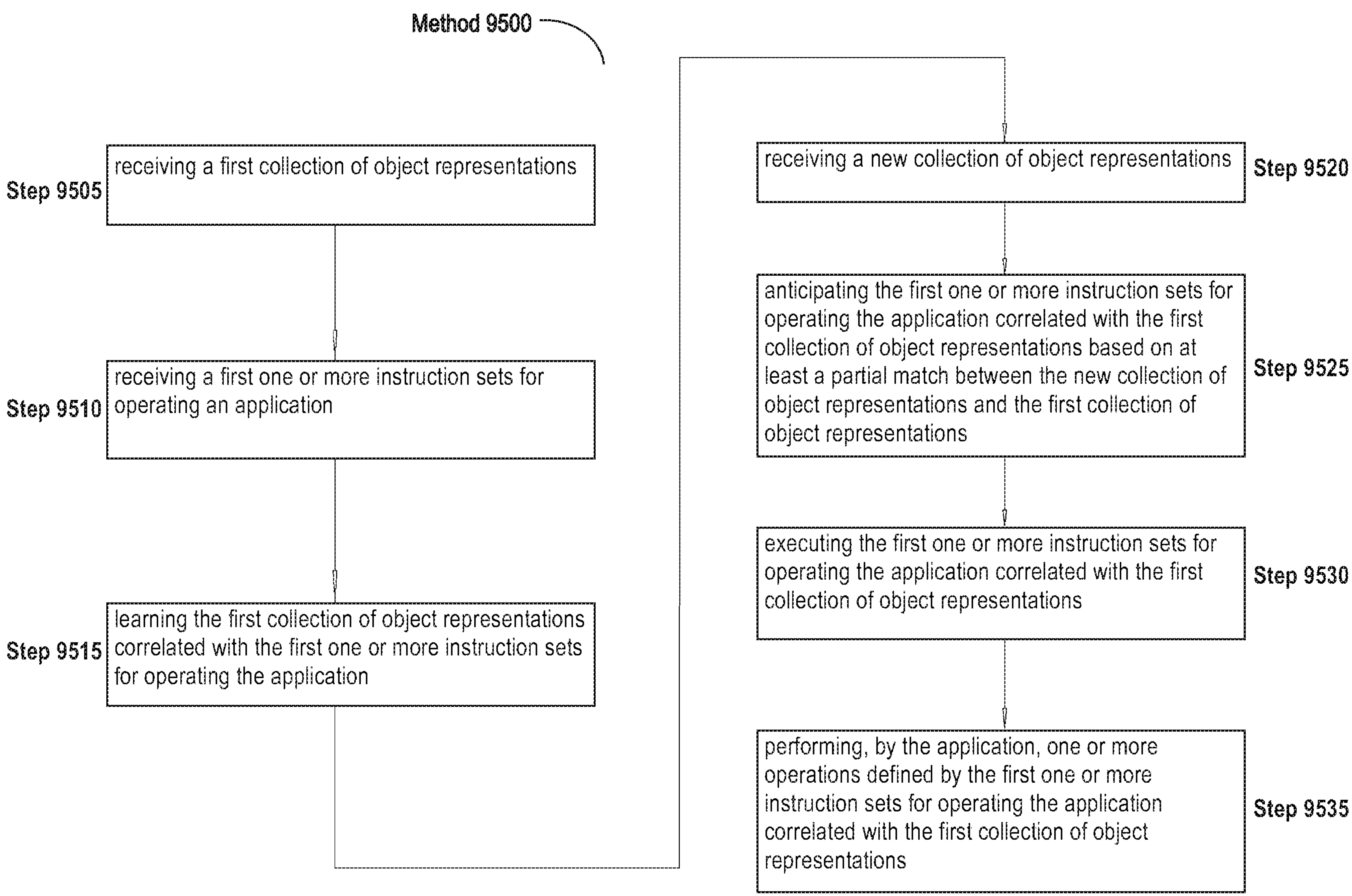

Method 9400
Step 9405
receiving a first collection of object representations
Step 9410
receiving a first one or more instruction sets for operating an application
Step 9415
correlating the first collection of object representations with the first one or more instruction sets for operating the application
Step 9420
storing the first collection of object representations correlated with the first one or more instruction sets for operating the application
Step 9425
receiving a new collection of object representations
Step 9430
comparing the new collection of object representations with the first collection of object representations
Step 9435
determining that there is at least a partial match between the new collection of object representations and the first collection of object representations
Step 9440
executing the first one or more instruction sets for operating the application correlated with the first collection of object representations
Step 9445
performing, by the application, one or more operations defined by the first one or more instruction sets for operating the application correlated with the first collection of object representations

FIG. 33



Method 9500
Step 9505
receiving a first collection of object representations
Step 9510
receiving a first one or more instruction sets for operating an application
Step 9515
learning the first collection of object representations correlated with the first one or more instruction sets for operating the application
Step 9520
receiving a new collection of object representations
Step 9525
anticipating the first one or more instruction sets for operating the application correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations
Step 9530
executing the first one or more instruction sets for operating the application correlated with the first collection of object representations
Step 9535
performing, by the application, one or more operations defined by the first one or more instruction sets for operating the application correlated with the first collection of object representations

FIG. 34

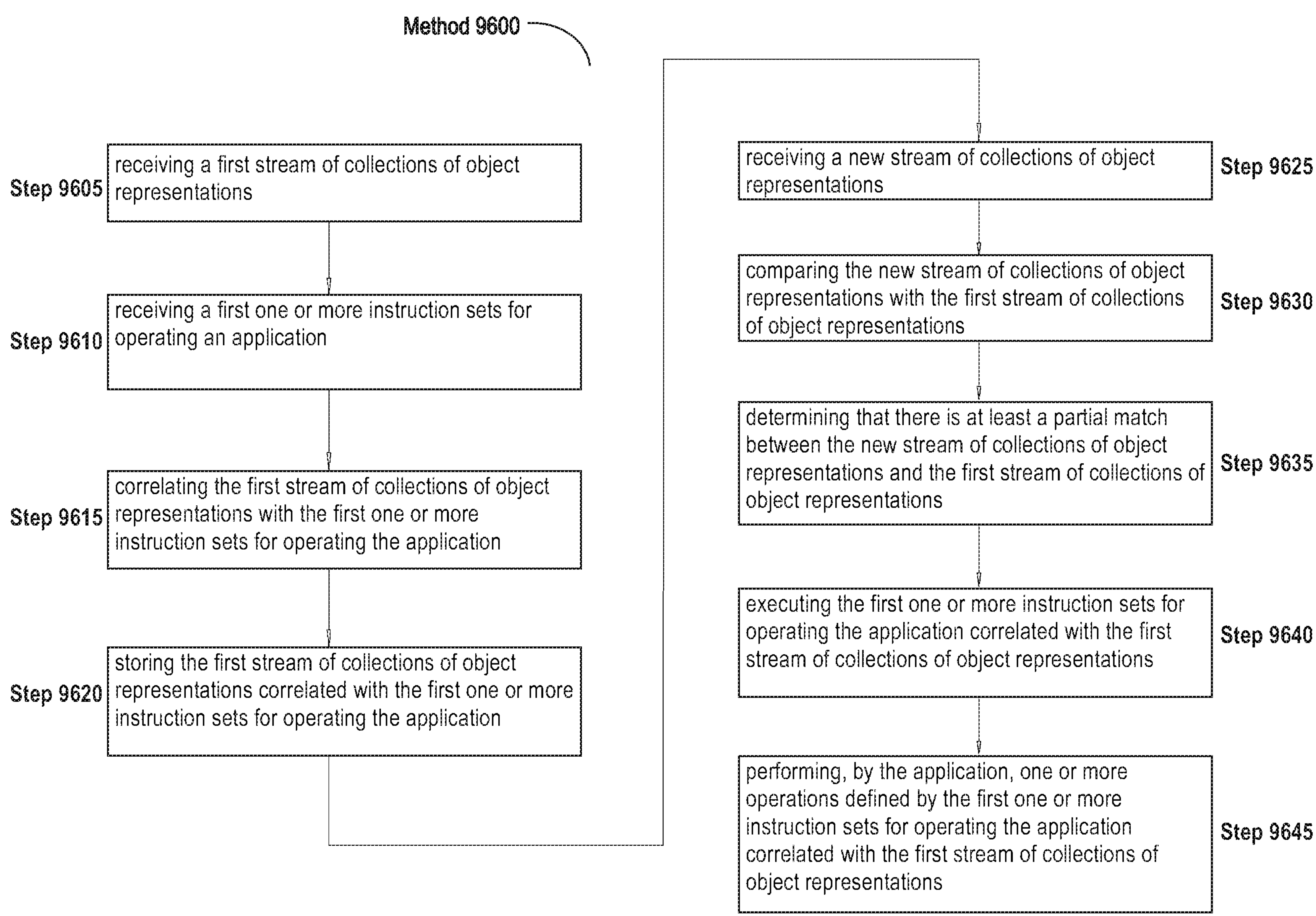

Method 9600
Step 9605
receiving a first stream of collections of object representations
Step 9610
receiving a first one or more instruction sets for operating an application
Step 9615
correlating the first stream of collections of object representations with the first one or more instruction sets for operating the application
Step 9620
storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the application
Step 9625
receiving a new stream of collections of object representations
Step 9630
comparing the new stream of collections of object representations with the first stream of collections of object representations
Step 9635
determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations
Step 9640
executing the first one or more instruction sets for operating the application correlated with the first stream of collections of object representations
Step 9645
performing, by the application, one or more operations defined by the first one or more instruction sets for operating the application correlated with the first stream of collections of object representations

FIG. 35

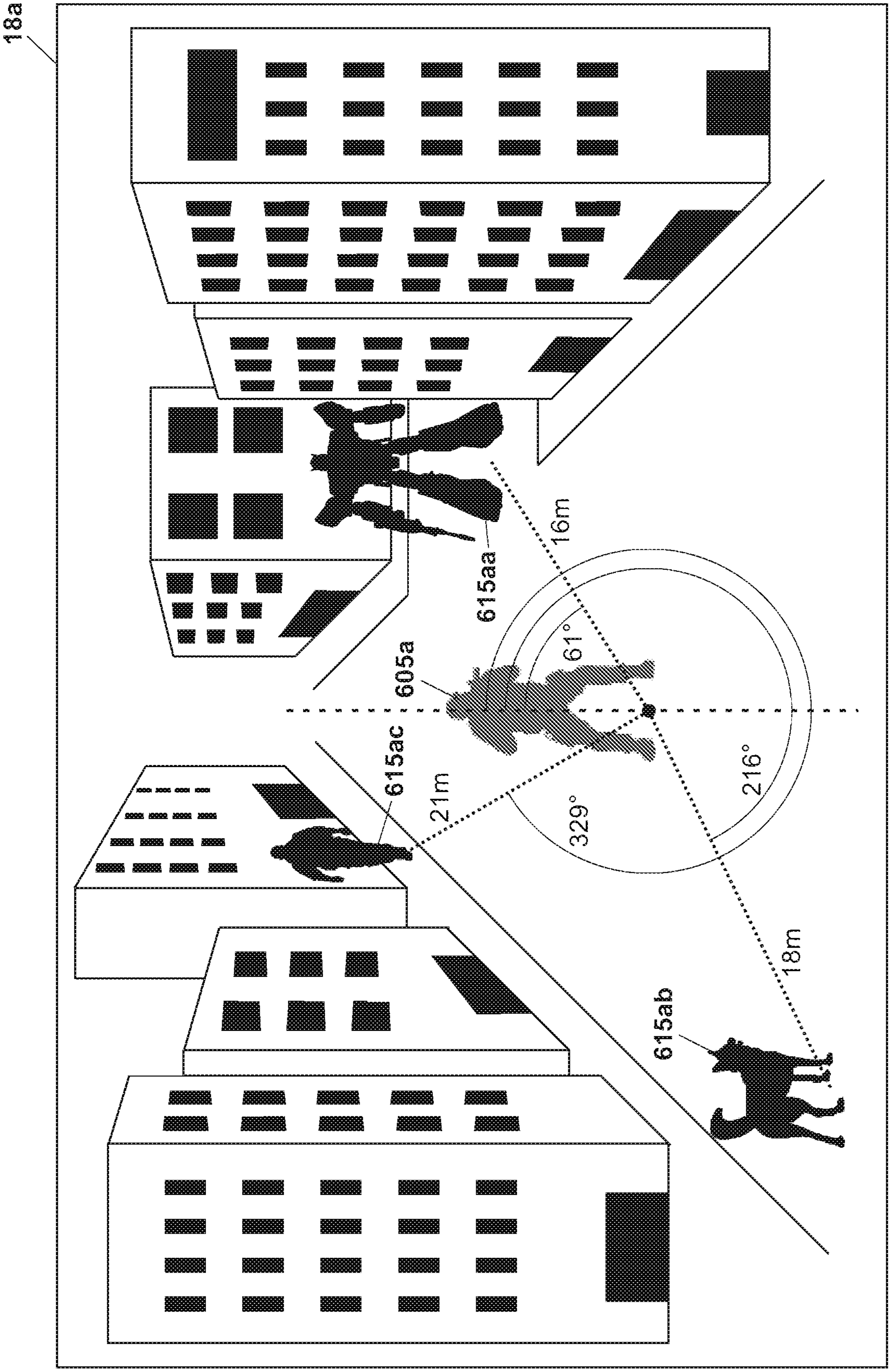
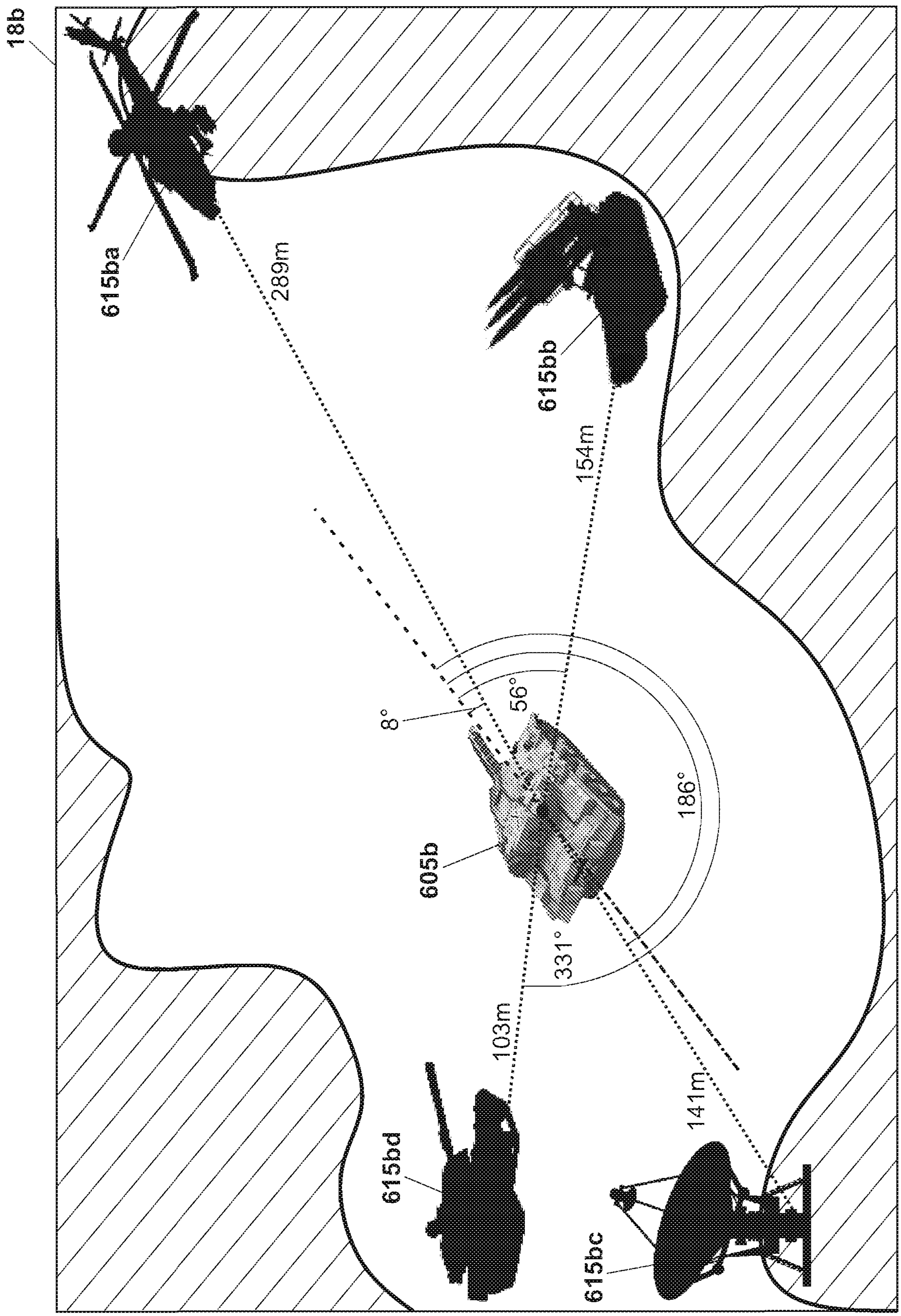


FIG. 36



18b
615ba
289m
615bb
154m
8°
56°
186°
605b
331°
103m
141m
615bd
615bc

FIG. 37

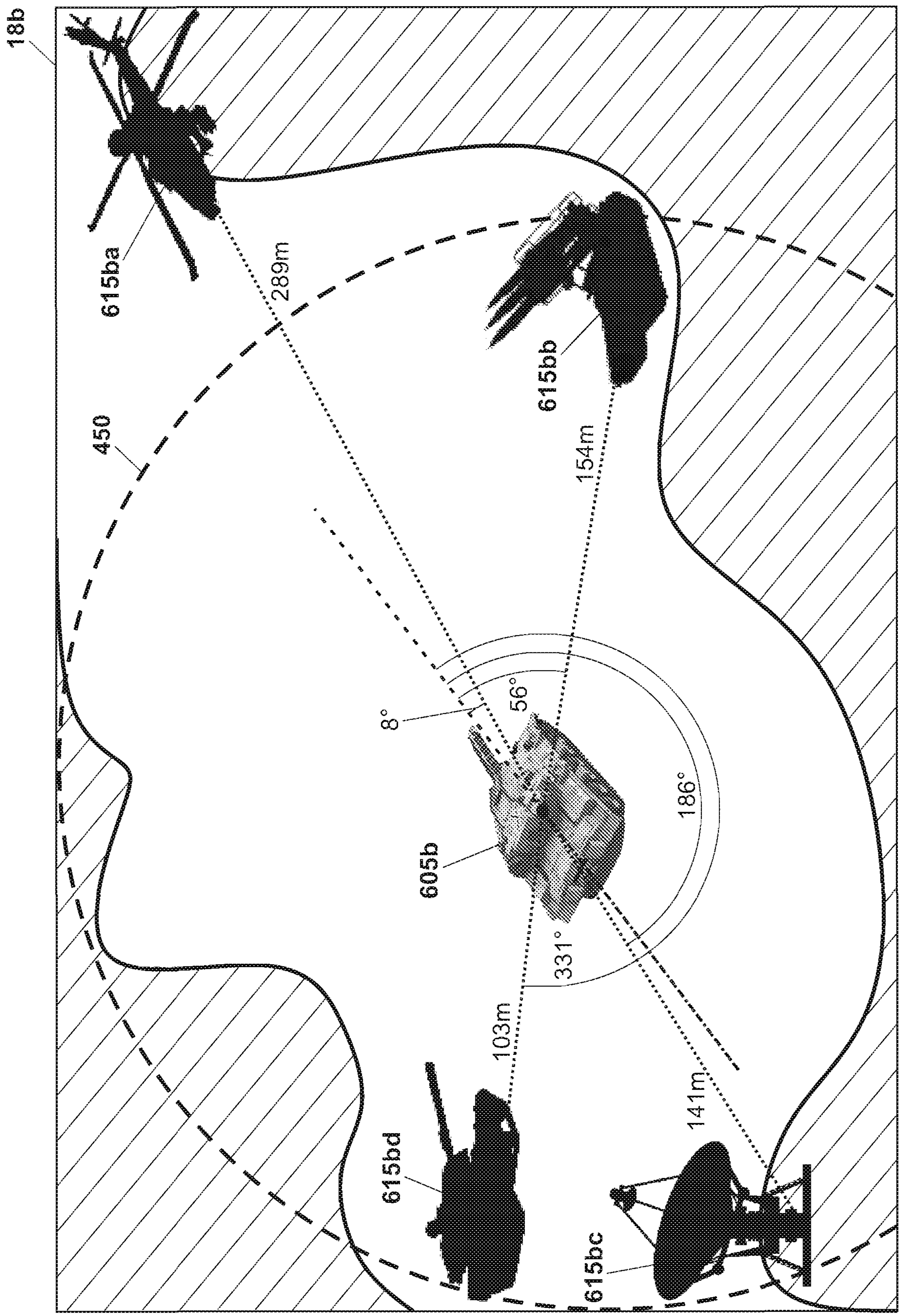

18b
615ba
289m
450
615bb
154m
8°
56°
186°
605b
331°
103m
141m
615bd
615bc

FIG. 38

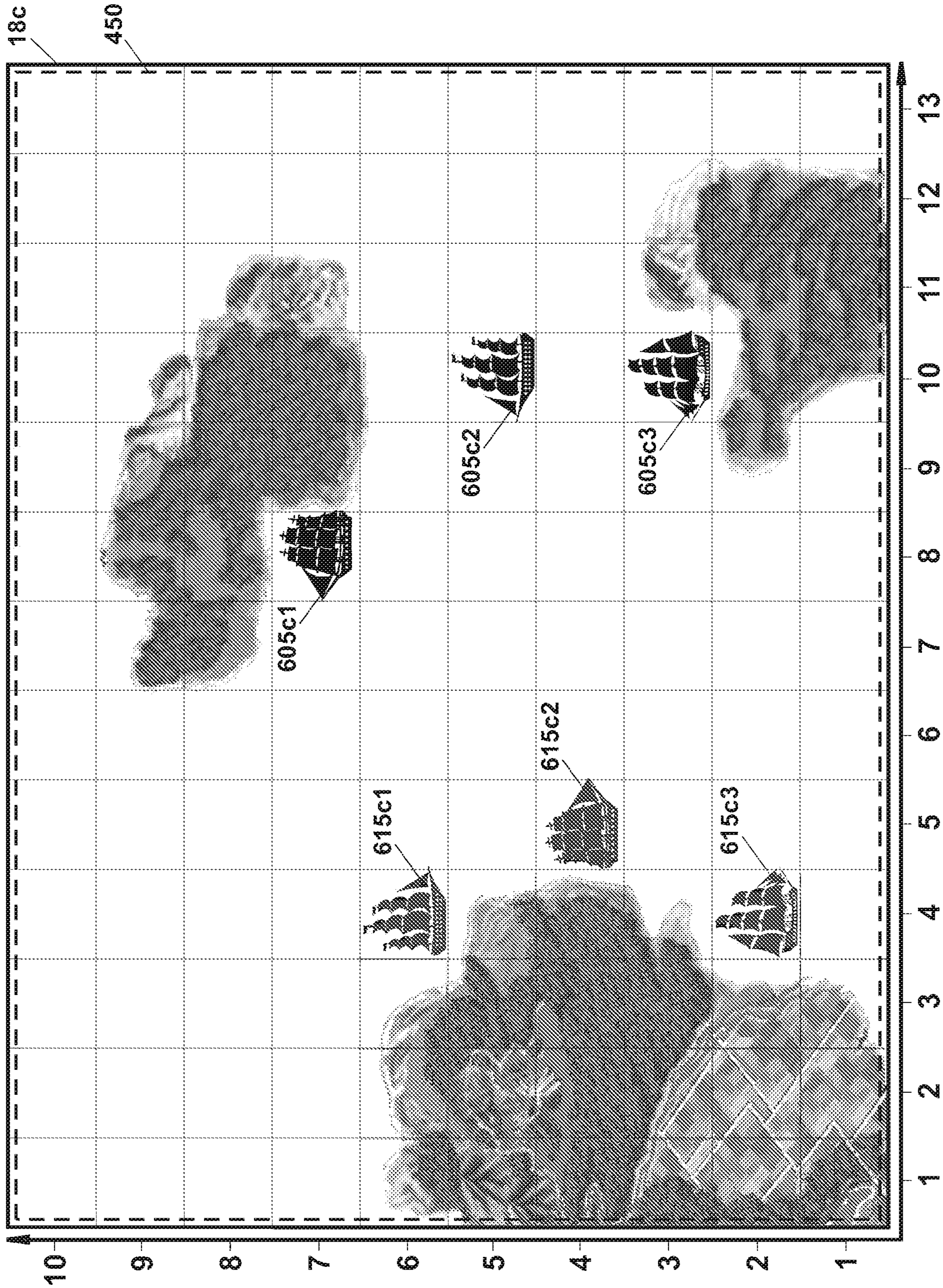

18c
450
605c1
605c2
605c3
615c1
615c2
615c3
1 2 3 4 5 6 7 8 9 10 11 12 13
1 2 3 4 5 6 7 8 9 10

FIG. 39

CLAIMS

Claim 1. A system for learning and using an avatar's circumstances for autonomous avatar operating, the system implemented at least in part on one or more computing devices, the system comprising:

a processor circuit configured to execute instruction sets of an application;

a memory unit configured to store data; and

an artificial intelligence unit configured to:

receive a first stream of collections of object representations, the first stream of collections of object representations including one or more object representations representing one or more objects of the application;

receive a first one or more instruction sets for operating an avatar of the application;

learn the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application;

receive a new stream of collections of object representations, the new stream of collections of object representations including one or more object representations representing one or more objects of the application;

anticipate the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of

collections of object representations and the first stream of collections of object representations; and

cause the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the avatar of the application performs one or more operations defined by the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations.

Claim 2. The system of Claim 1, wherein the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of the avatar of the application.

Claim 3. The system of Claim 1, wherein the receiving the first one or more instruction sets for operating the avatar of the application includes a tracing of the avatar of the application.

Claim 4. The system of Claim 1, wherein the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of the application.

Claim 5. The system of Claim 1, wherein the receiving the first one or more instruction sets for operating the avatar of the application includes a tracing of the application.

Claim 6. The system of Claim 1, wherein the receiving the first one or more instruction sets for operating the avatar of the application includes a tracing of the processor circuit or a tracing of a component of the processor circuit.

Claim 7. The system of Claim 1, wherein the first one or more instruction sets for operating the avatar of the application include one or more instruction sets that temporally correspond to the first stream of collections of object representations.

Claim 8. The system of Claim 1, wherein the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application includes storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application into the memory unit, the memory unit comprising a plurality of streams of collections of object representations correlated with one or more instruction sets for operating the avatar of the application.

Claim 9. The system of Claim 1, wherein the first stream of collections of object representations includes one or more collections of object representations, and wherein each collection of object representations of the first stream of collections of object representations includes one or more object representations.

Claim 10. The system of Claim 1, wherein the new stream of collections of object representations includes one or more collections of object representations, and wherein each collection of object representations of the new stream of collections of object representations includes one or more object representations.

Claim 11. The system of Claim 1, wherein at least one of: the processor circuit, the memory unit, or the artificial intelligence unit of the system are part of a single computing device.

Claim 12. The system of Claim 1, wherein the artificial intelligence unit is a hardware element that is part of, an application operating on, or an element coupled to the processor circuit.

Claim 13. A non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising:

receiving a first stream of collections of object representations, the first stream of collections of object representations including one or more object representations representing one or more objects of an application;

receiving a first one or more instruction sets for operating an avatar of the application;

learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application;

receiving a new stream of collections of object representations, the new stream of collections of object representations including one or more object representations representing one or more objects of the application;

anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations; and

causing an execution of the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations,

wherein the avatar of the application performs one or more operations defined by the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations.

Claim 14. The non-transitory computer storage medium of Claim 13, wherein the execution of the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations is performed by the one or more processor circuits or by another one or more processor circuits.

Claim 15. The non-transitory computer storage medium of Claim 13, wherein the receiving the first one or more instruction sets for operating the avatar of the application includes a tracing of the avatar of the application.

Claim 16. The non-transitory computer storage medium of Claim 13, wherein the receiving the first one or more instruction sets for operating the avatar of the application includes a tracing of the application.

Claim 17. A method comprising:

(a) receiving a first stream of collections of object representations by a processor circuit, the first stream of collections of object representations including one or more object representations representing one or more objects of an application;

(b) receiving a first one or more instruction sets for operating an avatar of the application by the processor circuit;

(c) learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application, the learning of (c) performed by the processor circuit;

(d) receiving a new stream of collections of object representations by the processor circuit, the new stream of collections of object representations including one or more object representations representing one or more objects of the application;

(e) anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, the anticipating of (e) performed by the processor circuit;

(f) executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations, the executing of (f) performed in response to the anticipating of (e); and

(g) performing, by the avatar of the application, one or more operations defined by the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations.

Claim 18. The method of Claim 17, wherein the executing of (f) is performed by the processor circuit or by another processor circuit.

Claim 19. The method of Claim 17, wherein the receiving of (b) includes a tracing of the avatar of the application.

Claim 20. The method of Claim 17, wherein the receiving of (b) includes a tracing of the application.

PTO/AIA/14 (11-15)
Approved for use through 04/30/2017. OMB 0651-0032
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	
	Application Number	
Title of Invention	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING AN AVATAR'S CIRCUMSTANCES FOR AUTONOMOUS AVATAR OPERATION	

The application data sheet is part of the provisional or nonprovisional application for which it is being submitted. The following form contains the bibliographic data arranged in a format specified by the United States Patent and Trademark Office as outlined in 37 CFR 1.76.
This document may be completed electronically and submitted to the Office in electronic format using the Electronic Filing System (EFS) or the document may be printed and included in a paper filed application.

Secrecy Order 37 CFR 5.2:

☐ Portions or all of the application associated with this Application Data Sheet may fall under a Secrecy Order pursuant to 37 CFR 5.2 (Paper filers only. Applications that fall under Secrecy Order may not be filed electronically.)

Inventor Information:

Inventor 1 Remove

Legal Name

Prefix	Given Name	Middle Name	Family Name	Suffix
	Jasmin		Cosic	

Residence Information (Select One) ● US Residency Non US Residency Active US Military Service

City	Miami	State/Province	FL	Country of Residence	US

Mailing Address of Inventor:

Address 1	108 Woodbury Street		
Address 2			
City	Pawtucket	**State/Province**	RI
Postal Code	02861	**Country**	US

All Inventors Must Be Listed - Additional Inventor Information blocks may be generated within this form by selecting the **Add** button. Add

Correspondence Information:

Enter either Customer Number or complete the Correspondence Information section below. For further information see 37 CFR 1.33(a).

☐ **An Address is being provided for the correspondence Information of this application.**

Customer Number	116094
Email Address	cpapc29@hotmail.com

Add Email Remove Email

Application Information:

Title of the Invention	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING AN AVATAR'S CIRCUMSTANCES FOR AUTONOMOUS AVATAR OPERATION		
Attorney Docket Number		**Small Entity Status Claimed**	☒
Application Type	Nonprovisional		
Subject Matter	Utility		
Total Number of Drawing Sheets (if any)	39	**Suggested Figure for Publication (if any)**	2

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	
	Application Number	
Title of Invention	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING AN AVATAR'S CIRCUMSTANCES FOR AUTONOMOUS AVATAR OPERATION	

Filing By Reference:

Only complete this section when filing an application by reference under 35 U.S.C. 111(c) and 37 CFR 1.57(a). Do not complete this section if application papers including a specification and any drawings are being filed. Any domestic benefit or foreign priority information must be provided in the appropriate section(s) below (i.e., "Domestic Benefit/National Stage Information" and "Foreign Priority Information").

For the purposes of a filing date under 37 CFR 1.53(b), the description and any drawings of the present application are replaced by this reference to the previously filed application, subject to conditions and requirements of 37 CFR 1.57(a).

Application number of the previously filed application	Filing date (YYYY-MM-DD)	Intellectual Property Authority or Country

Publication Information:

☐ Request Early Publication (Fee required at time of Request 37 CFR 1.219)

☒ **Request Not to Publish.** I hereby request that the attached application not be published under 35 U.S.C. 122(b) and certify that the invention disclosed in the attached application **has not and will not** be the subject of an application filed in another country, or under a multilateral international agreement, that requires publication at eighteen months after filing.

Representative Information:

Representative information should be provided for all practitioners having a power of attorney in the application. Providing this information in the Application Data Sheet does not constitute a power of attorney in the application (see 37 CFR 1.32).
Either enter Customer Number or complete the Representative Name section below. If both sections are completed the customer Number will be used for the Representative Information during processing.

Please Select One:	● Customer Number	US Patent Practitioner	◯ Limited Recognition (37 CFR 11.9)
Customer Number			

Domestic Benefit/National Stage Information:

This section allows for the applicant to either claim benefit under 35 U.S.C. 119(e), 120, 121, 365(c), or 386(c) or indicate National Stage entry from a PCT application. Providing benefit claim information in the Application Data Sheet constitutes the specific reference required by 35 U.S.C. 119(e) or 120, and 37 CFR 1.78.
When referring to the current application, please leave the "Application Number" field blank.

Prior Application Status			Remove
Application Number	Continuity Type	Prior Application Number	Filing or 371(c) Date (YYYY-MM-DD)

Additional Domestic Benefit/National Stage Data may be generated within this form by selecting the **Add** button. Add

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	
	Application Number	

Title of Invention	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING AN AVATAR'S CIRCUMSTANCES FOR AUTONOMOUS AVATAR OPERATION

Foreign Priority Information:

This section allows for the applicant to claim priority to a foreign application. Providing this information in the application data sheet constitutes the claim for priority as required by 35 U.S.C. 119(b) and 37 CFR 1.55. When priority is claimed to a foreign application that is eligible for retrieval under the priority document exchange program (PDX)[i] the information will be used by the Office to automatically attempt retrieval pursuant to 37 CFR 1.55(i)(1) and (2). Under the PDX program, applicant bears the ultimate responsibility for ensuring that a copy of the foreign application is received by the Office from the participating foreign intellectual property office, or a certified copy of the foreign priority application is filed, within the time period specified in 37 CFR 1.55(g)(1).

Application Number	Country[i]	Filing Date (YYYY-MM-DD)	Remove Access Code[i] (if applicable)

Additional Foreign Priority Data may be generated within this form by selecting the **Add** button. Add

Statement under 37 CFR 1.55 or 1.78 for AIA (First Inventor to File) Transition Applications

☐ This application (1) claims priority to or the benefit of an application filed before March 16, 2013 and (2) also contains, or contained at any time, a claim to a claimed invention that has an effective filing date on or after March 16, 2013.
NOTE: By providing this statement under 37 CFR 1.55 or 1.78, this application, with a filing date on or after March 16, 2013, will be examined under the first inventor to file provisions of the AIA.

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	
	Application Number	

Title of Invention	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING AN AVATAR'S CIRCUMSTANCES FOR AUTONOMOUS AVATAR OPERATION

Authorization or Opt-Out of Authorization to Permit Access:

When this Application Data Sheet is properly signed and filed with the application, applicant has provided written authority to permit a participating foreign intellectual property (IP) office access to the instant application-as-filed (see paragraph A in subsection 1 below) and the European Patent Office (EPO) access to any search results from the instant application (see paragraph B in subsection 1 below).

Should applicant choose not to provide an authorization identified in subsection 1 below, applicant **must opt-out** of the authorization by checking the corresponding box A or B or both in subsection 2 below.

NOTE: This section of the Application Data Sheet is **ONLY** reviewed and processed with the **INITIAL** filing of an application. After the initial filing of an application, an Application Data Sheet cannot be used to provide or rescind authorization for access by a foreign IP office(s). Instead, Form PTO/SB/39 or PTO/SB/69 must be used as appropriate.

1. Authorization to Permit Access by a Foreign Intellectual Property Office(s)

A. Priority Document Exchange (PDX) - Unless box A in subsection 2 (opt-out of authorization) is checked, the undersigned hereby **grants the USPTO authority** to provide the European Patent Office (EPO), the Japan Patent Office (JPO), the Korean Intellectual Property Office (KIPO), the State Intellectual Property Office of the People's Republic of China (SIPO), the World Intellectual Property Organization (WIPO), and any other foreign intellectual property office participating with the USPTO in a bilateral or multilateral priority document exchange agreement in which a foreign application claiming priority to the instant patent application is filed, access to: (1) the instant patent application-as-filed and its related bibliographic data, (2) any foreign or domestic application to which priority or benefit is claimed by the instant application and its related bibliographic data, and (3) the date of filing of this Authorization. See 37 CFR 1.14(h)(1).

B. Search Results from U.S. Application to EPO - Unless box B in subsection 2 (opt-out of authorization) is checked, the undersigned hereby **grants the USPTO authority** to provide the EPO access to the bibliographic data and search results from the instant patent application when a European patent application claiming priority to the instant patent application is filed. See 37 CFR 1.14(h)(2).

The applicant is reminded that the EPO's Rule 141(1) EPC (European Patent Convention) requires applicants to submit a copy of search results from the instant application without delay in a European patent application that claims priority to the instant application.

2. Opt-Out of Authorizations to Permit Access by a Foreign Intellectual Property Office(s)

☒ A. Applicant **DOES NOT** authorize the USPTO to permit a participating foreign IP office access to the instant application-as-filed. If this box is checked, the USPTO will not be providing a participating foreign IP office with any documents and information identified in subsection 1A above.

☒ B. Applicant **DOES NOT** authorize the USPTO to transmit to the EPO any search results from the instant patent application. If this box is checked, the USPTO will not be providing the EPO with search results from the instant application.

NOTE: Once the application has published or is otherwise publicly available, the USPTO may provide access to the application in accordance with 37 CFR 1.14.

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	
	Application Number	
Title of Invention	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING AN AVATAR'S CIRCUMSTANCES FOR AUTONOMOUS AVATAR OPERATION	

Applicant Information:

Providing assignment information in this section does not substitute for compliance with any requirement of part 3 of Title 37 of CFR to have an assignment recorded by the Office.

Applicant 1 Remove

If the applicant is the inventor (or the remaining joint inventor or inventors under 37 CFR 1.45), this section should not be completed. The information to be provided in this section is the name and address of the legal representative who is the applicant under 37 CFR 1.43; or the name and address of the assignee, person to whom the inventor is under an obligation to assign the invention, or person who otherwise shows sufficient proprietary interest in the matter who is the applicant under 37 CFR 1.46. If the applicant is an applicant under 37 CFR 1.46 (assignee, person to whom the inventor is obligated to assign, or person who otherwise shows sufficient proprietary interest) together with one or more joint inventors, then the joint inventor or inventors who are also the applicant should be identified in this section.

Clear

Assignee	Legal Representative under 35 U.S.C. 117	Joint Inventor
Person to whom the inventor is obligated to assign.	Person who shows sufficient proprietary interest	

If applicant is the legal representative, indicate the authority to file the patent application, the inventor is:

Name of the Deceased or Legally Incapacitated Inventor:

If the Applicant is an Organization check here. ☐

Prefix	**Given Name**	Middle Name	**Family Name**	Suffix

Mailing Address Information For Applicant:

Address 1			
Address 2			
City		**State/Province**	
Country		Postal Code	
Phone Number		Fax Number	
Email Address			

Additional Applicant Data may be generated within this form by selecting the Add button. Add

Assignee Information including Non-Applicant Assignee Information:

Providing assignment information in this section does not substitute for compliance with any requirement of part 3 of Title 37 of CFR to have an assignment recorded by the Office.

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	
	Application Number	
Title of Invention	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING AN AVATAR'S CIRCUMSTANCES FOR AUTONOMOUS AVATAR OPERATION	

Assignee 1

Complete this section if assignee information, including non-applicant assignee information, is desired to be included on the patent application publication. An assignee-applicant identified in the "Applicant Information" section will appear on the patent application publication as an applicant. For an assignee-applicant, complete this section only if identification as an assignee is also desired on the patent application publication.

Remove

If the Assignee or Non-Applicant Assignee is an Organization check here. ☐

Prefix	**Given Name**	Middle Name	**Family Name**	Suffix

Mailing Address Information For Assignee including Non-Applicant Assignee:

Address 1			
Address 2			
City		**State/Province**	
Country i		Postal Code	
Phone Number		Fax Number	
Email Address			

Additional Assignee or Non-Applicant Assignee Data may be generated within this form by selecting the Add button. Add

Signature:

Remove

NOTE: This Application Data Sheet must be signed in accordance with 37 CFR 1.33(b). **However, if this Application Data Sheet is submitted with the <u>INITIAL</u> filing of the application <u>and</u> either box A or B is <u>not</u> checked in subsection 2 of the "Authorization or Opt-Out of Authorization to Permit Access" section, then this form must also be signed in accordance with 37 CFR 1.14(c).**

This Application Data Sheet **<u>must</u>** be signed by a patent practitioner if one or more of the applicants is a **juristic entity** (e.g., corporation or association). If the applicant is two or more joint inventors, this form must be signed by a patent practitioner, **<u>all</u>** joint inventors who are the applicant, or one or more joint inventor-applicants who have been given power of attorney (e.g., see USPTO Form PTO/AIA/81) on behalf of **<u>all</u>** joint inventor-applicants.

See 37 CFR 1.4(d) for the manner of making signatures and certifications.

Signature	/Jasmin Cosic/			Date (YYYY-MM-DD)	2016-12-19
First Name	Jasmin	Last Name	Cosic	Registration Number	

Additional Signature may be generated within this form by selecting the Add button. Add

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	
	Application Number	

Title of Invention	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING AN AVATAR'S CIRCUMSTANCES FOR AUTONOMOUS AVATAR OPERATION

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1 The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these records.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3 A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent CooperationTreaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

ABSTRACT

Aspects of the disclosure generally relate to computing devices and/or systems, and may be generally directed to devices, systems, methods, and/or applications for learning an avatar's or an application's operation in various circumstances, storing this knowledge in a knowledgebase (i.e. neural network, graph, sequences, etc.), and/or enabling autonomous operation of the avatar or the application.

ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING AN AVATAR'S CIRCUMSTANCES FOR AUTONOMOUS AVATAR OPERATION

FIELD

The disclosure generally relates to computing devices and/or systems. The disclosure includes devices, apparatuses, systems, and related methods for providing advanced learning, anticipating, decision making, automation, and/or other functionalities.

COPYRIGHT NOTICE

BACKGROUND

Applications and/or avatars thereof commonly operate by receiving a user's operating directions in various circumstances. Instructions are then executed to effect the operation of an application and/or avatar based on user's operating directions. Hence, applications and/or avatars rely on the user to direct their behaviors. Commonly employed application and/or avatar operating techniques lack a way to learn operation of an application and/or avatar and enable autonomous operation of an application and/or avatar.

SUMMARY

In some aspects, the disclosure relates to a system for learning and using an avatar's circumstances for autonomous avatar operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets of an application. The system may further comprise: a memory unit configured to store data. The system may further comprise: an artificial intelligence unit configured to. The artificial intelligence unit may be configured to: receive a first collection of object representations, the first collection of object representations including one or more object representations representing one or more objects of the application. The artificial intelligence unit may be further configured to: receive a first one or more instruction sets for operating an avatar of the application. The artificial intelligence unit may be further configured to: learn the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application. The artificial intelligence unit may be further configured to: receive a new collection of object representations, the new collection of object representations including one or more object representations representing one or more objects of the application. The artificial intelligence unit may be further configured to: anticipate the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations, the

causing performed in response to the anticipating of the artificial intelligence unit, wherein the avatar of the application performs one or more operations defined by the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations.

In certain embodiments, the processor circuit includes one or more processor circuits. In further embodiments, the application includes a computer game, a virtual world, a 3D graphics application, a 2D graphics application, a web browser, a media application, a word processing application, a spreadsheet application, a database application, a forms-based application, an operating system, a device control application, a system control application, or a computer application. In further embodiments, at least one of: the processor circuit, the memory unit, or the artificial intelligence unit of the system are part of a single computing device.

In some embodiments, the memory unit includes one or more memory units. In further embodiments, the memory unit resides on a remote computing device or a remote computing system. The remote computing device or the remote computing system may include a server, a cloud, a computing device, or a computing system accessible over a network or an interface.

In certain embodiments, the artificial intelligence unit includes a circuit, a computing apparatus, a computing system, or a hardware element. In further embodiments, the artificial intelligence unit includes an application. In further embodiments, the artificial intelligence unit is coupled to the memory unit. In further embodiments, the artificial intelligence unit is a hardware element that is part of, an application operating on, or an element coupled to the processor circuit. In further embodiments, the artificial intelligence unit is part of or coupled to the application. In further embodiments, the artificial intelligence unit is part of or coupled to the avatar of the application. In further embodiments, the system further comprises: an additional processor circuit, wherein the artificial intelligence unit is a hardware element that is part of, an application operating on, or an element coupled to the additional processor circuit. In further embodiments, the artificial intelligence unit is a hardware element that is part of, an application operating on, or an element coupled to a remote computing device or a remote computing system. In further embodiments, the artificial intelligence unit is attachable to the processor circuit. In further embodiments, the artificial intelligence unit is attachable to the application. In further embodiments, the artificial intelligence unit is attachable to the avatar of the application. In further embodiments, the artificial intelligence unit is embedded or built into the processor circuit. In further embodiments, the artificial intelligence unit is embedded or built into the application. In further embodiments, the artificial intelligence unit is embedded or built into the avatar of the application. In further embodiments, the artificial intelligence unit is provided as a feature of the processor circuit. In further embodiments, the artificial intelligence unit is provided as a feature of the application. In further embodiments, the artificial intelligence unit is provided as a feature of the avatar of the application. In further embodiments, the artificial intelligence unit is further configured to: take control from, share control with, or release control to the processor circuit. In further embodiments, the artificial intelligence unit is further configured to: take control from, share control with, or release control to the application. In further embodiments, the artificial intelligence unit is further configured to: take control from, share control with, or release control to the avatar of the application.

In some embodiments, the one or more objects of the application include a 2D model, a 3D model, a 2D shape, a 3D shape, a graphical user interface element, a form element, a data or database element, a spreadsheet element, a link, a picture, a text, a number, or a computer object. In further embodiments, the one or more objects of the application include one or more objects of the application in the avatar's surrounding. The avatar's

surrounding may include an area of interest around the avatar. In further embodiments, the avatar of the application includes a user-controllable object of the application. In further embodiments, an avatar's circumstance includes one or more objects of the application.

In certain embodiments, the first collection of object representations is received at a first time. In further embodiments, the new collection of object representations is received at a new time. In further embodiments, the first collection of object representations includes a unit of knowledge of the avatar's circumstance at a first time. In further embodiments, the new collection of object representations includes a unit of knowledge of the avatar's circumstance at a new time. In further embodiments, an object representation includes one or more properties of an object of the application. In further embodiments, an object representation includes one or more information on an object of the application. In further embodiments, the first or the new collection of object representations includes or is associated with a time stamp, an order, or a time related information. In further embodiments, the first collection of object representations includes a comparative collection of object representations whose at least one portion can be used for comparisons with at least one portion of collections of object representations subsequent to the first collection of object representations, the collections of object representations subsequent to the first collection of object representations comprising the new collection of object representations. In further embodiments, the first collection of object representations includes a comparative collection of object representations that can be used for comparison with the new collection of object representations. In further embodiments, the new collection of object representations includes an anticipatory collection of object representations that can be compared with collections of object representations whose correlated one or more instruction sets for operating the avatar of the application can be used for anticipation of one or more instruction sets to be executed in autonomous operating of the avatar of the application. In further embodiments, the first collection of object representations includes a stream of collections of object representations. In further embodiments, the new collection of object representations includes a stream of collections of object representations.

In some embodiments, the receiving the first collection of object representations includes receiving one or more properties of the one or more objects of the application. The one or more properties of the one or more objects of the application may include one or more information on the one or more objects of the application. The receiving the one or more properties of the one or more objects of the application may include receiving the one or more properties of the one or more objects of the application from an engine, an environment, or a system used to implement the application. The receiving the one or more properties of the one or more objects of the application may include at least one of: accessing or reading a scene graph or a data structure used for organizing the one or more objects of the application. The receiving the one or more properties of the one or more objects of the application may include detecting the one or more properties of the one or more objects of the application in a picture of the avatar's surrounding. The receiving the one or more properties of the one or more objects of the application may include detecting the one or more properties of the one or more objects of the application in a sound from the avatar's surrounding.

In certain embodiments, the system further comprises: an object processing unit configured to receive collections of object representations, wherein the first or the new collection of object representations is received by the object processing unit.

In some embodiments, the first one or more instruction sets for operating the avatar of the application include one or more instruction sets that temporally correspond to the first collection of object representations. The one or more instruction sets that temporally correspond to the first collection of object representations may include one or more instruction sets executed at a time of generating the first collection of object representations. The one or more instruction sets that temporally correspond to the first collection of object representations may include one or more instruction sets executed prior to generating the first collection of object representations. The one or more instruction sets that temporally correspond to the first collection of object representations may include one or more instruction sets executed within a threshold period of time prior to generating the first collection of object representations. The one or more instruction sets that temporally correspond to the first collection of object representations may include one or more instruction sets executed subsequent to generating the first collection of object representations. The one or more instruction sets that temporally correspond to the first collection of object representations may include one or more instruction sets executed within a threshold period of time subsequent to generating the first collection of object representations. The one or more instruction sets that temporally correspond to the first collection of object representations may include one or more instruction sets executed within a threshold period of time prior to generating the first collection of object representations and a threshold period of time subsequent to generating the first collection of object representations.

In certain embodiments, the first one or more instruction sets for operating the avatar of the application include one or more instruction sets executed in operating the avatar of the application. In further embodiments, the first one or more instruction sets for operating the avatar of the application are part of the application. In further embodiments, the first one or more instruction sets for operating the avatar of the application are part of the avatar of the application. In further embodiments, the first one or more instruction sets for operating the avatar of the application include one or more inputs into or one or more outputs from the processor circuit. In further embodiments, the first one or more instruction sets for operating the avatar of the application include a value or a state of a register or an element of the processor circuit. In further embodiments, the first one or more instruction sets for operating the avatar of the application include at least one of: a command, a keyword, a symbol, an instruction, an operator, a variable, a value, an object, a data structure, a function, a parameter, a state, a signal, an input, an output, a character, a digit, or a reference thereto. In further embodiments, the first one or more instruction sets for operating the avatar of the application include a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, a structured query language (SQL) code, or a machine code. In further embodiments, the first one or more instruction sets for operating the avatar of the application include one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the first one or more instruction sets for operating the avatar of the application include one or more instruction sets for operating the application.

In some embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes receiving the first one or more instruction sets for operating the avatar of the application executed by the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes receiving the first one or more instruction sets for operating the avatar of the application as they are executed by the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes obtaining the first one or more

instruction sets for operating the avatar of the application from the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes receiving the first one or more instruction sets for operating the avatar of the application from a register or an element of the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes receiving the first one or more instruction sets for operating the avatar of the application from at least one of: the memory unit, a virtual machine, a runtime engine, a hard drive, a storage device, a peripheral device, a network connected device, or a user. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes receiving the first one or more instruction sets for operating the avatar of the application from a plurality of processor circuits, applications, memory units, virtual machines, runtime engines, hard drives, storage devices, peripheral devices, network connected devices, or users. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes obtaining the first one or more instruction sets for operating the avatar of the application from the application. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes obtaining the first one or more instruction sets for operating the avatar of the application from the avatar of the application. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes receiving the one or more instruction sets for operating the avatar of the application at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: tracing, profiling, or instrumentation of a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, a structured query language (SQL) code, or a machine code. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: tracing, profiling, or instrumentation of a register of the processor circuit, the memory unit, a storage, or a repository where the first one or more instruction sets for operating the avatar of the application are stored. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: tracing, profiling, or instrumentation of the processor circuit, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a processing element. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: tracing, profiling, or instrumentation of the processor circuit or tracing, profiling, or instrumentation of a component of the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: tracing, profiling, or instrumentation of the application. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: tracing, profiling, or instrumentation of the avatar of the application. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: tracing, profiling, or instrumentation at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: tracing, profiling, or instrumentation of one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes

at least one of: tracing, profiling, or instrumentation of a user input. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: a manual, an automatic, a dynamic, or a just in time (JIT) tracing, profiling, or instrumentation. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes utilizing at least one of: a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, a logging tool, or an independent tool for obtaining instruction sets. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes utilizing an assembly language. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes utilizing a branch or a jump. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes a branch tracing or a simulation tracing. In further embodiments, the system further comprises: an interface configured to receive instruction sets, wherein the first one or more instruction sets for operating the avatar of the application are received via the interface. The interface may include an acquisition interface.

In certain embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application include a unit of knowledge of how the avatar of the application operated in a circumstance. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application are included in a neuron, a node, a vertex, or an element of a knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application are structured into a knowledge cell. In further embodiments, the knowledge cell is included in a neuron, a node, a vertex, or an element of a knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application includes correlating the first collection of object representations with the first one or more instruction sets for operating the avatar of the application. The correlating the first collection of object representations with the first one or more instruction sets for operating the avatar of the application may include generating a knowledge cell, the knowledge cell comprising the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application. The correlating the first collection of object representations with the first one or more instruction sets for operating the avatar of the application may include structuring a unit of knowledge of how the avatar of the application operated in a circumstance. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application includes learning a user's knowledge, style, or methodology of operating the avatar of the application in a circumstance.

In some embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application includes storing the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application into the memory unit, the memory unit comprising a plurality of collections of object representations correlated with one

or more instruction sets for operating the avatar of the application. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the avatar of the application include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the avatar of the application are organized into a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, one or more collection of object representations correlated with one or more instruction sets for operating the avatar of the application of the plurality of collections of object representations correlated with one or more instruction sets for operating the avatar of the application are included in one or more neurons, nodes, vertices, or elements of a knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the avatar of the application include a user's knowledge, style, or methodology of operating the avatar of the application in circumstances. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the avatar of the application are stored on a remote computing device or a remote computing system. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the avatar of the application include an artificial intelligence system for knowledge structuring, storing, or representation. The artificial intelligence system for knowledge structuring, storing, or representation may include at least one of: a deep learning system, a supervised learning system, an unsupervised learning system, a neural network, a search-based system, an optimization-based system, a logic-based system, a fuzzy logic-based system, a tree-based system, a graph-based system, a hierarchical system, a symbolic system, a sub-symbolic system, an evolutionary system, a genetic system, a multi-agent system, a deterministic system, a probabilistic system, or a statistical system.

In certain embodiments, the anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes comparing at least one portion of the new collection of object representations with at least one portion of the first collection of object representations. The at least one portion of the new collection of object representations may include at least one object representation or at least one object property of the new collection of object representations. The at least one portion of the first collection of object representations may include at least one object representation or at least one object property of the first collection of object representations. In further embodiments, the anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes comparing at least one object representation from the new collection of object representations with at least one object representation from the first collection of object representations. In further embodiments, the comparing at least one object representation from the new collection of object representations with at least one object representation from the first collection of object representations includes

comparing at least one object property of the at least one object representation from the new collection of object representations with at least one object property of the at least one object representation from the first collection of object representations. In further embodiments, the anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes comparing at least one object property of at least one object representation from the new collection of object representations with at least one object property of at least one object representation from the first collection of object representations.

In some embodiments, the anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes determining that there is at least a partial match between the new collection of object representations and the first collection of object representations. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that there is at least a partial match between one or more portions of the new collection of object representations and one or more portions of the first collection of object representations. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that a similarity between at least one portion of the new collection of object representations and at least one portion of the first collection of object representations exceeds a similarity threshold. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining a substantial similarity between at least one portion of the new collection of object representations and at least one portion of the first collection of object representations. The substantial similarity may be achieved when a similarity between the at least one portion of the new collection of object representations and the at least one portion of the first collection of object representations exceeds a similarity threshold. The substantial similarity may be achieved when a number or a percentage of matching or partially matching portions of the new collection of object representations and portions of the first collection of object representations exceeds a threshold number or threshold percentage. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that a number or a percentage of matching or partially matching object representations from the new collection of object representations and from the first collection of object representations exceeds a threshold number or threshold percentage. The matching or partially matching object representations from the new collection of object representations and from the first collection of object representations may be determined factoring in at least one of: a type of an object representation, an importance of an object representation, a threshold for a similarity in an object representation, or a threshold for a difference in an object representation. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that a number or a percentage of matching or partially matching object properties from the new collection of object representations and from the first collection of object representations exceeds a threshold number or threshold percentage. The matching or partially matching object properties from the new collection of

object representations and from the first collection of object representations may be determined factoring in at least one of: an association of an object property with an object representation, a category of an object property, an importance of an object property, a threshold for a similarity in an object property, or a threshold for a difference in an object property. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that there is at least a partial match between at least one object representation from the new collection of object representations and at least one object representation from the first collection of object representations. The determining that there is at least a partial match between at least one object representation from the new collection of object representations and at least one object representation from the first collection of object representations may include determining that there is at least a partial match between at least one object property of the at least one object representation from the new collection of object representations and at least one object property of the at least one object representation from the first collection of object representations.

In certain embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations instead of or prior to an instruction set that would have been executed next. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying one or more instruction sets of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes inserting the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations into a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes redirecting the processor circuit to the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes redirecting the processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes transmitting, to the processor circuit for execution, the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes issuing an interrupt to the processor circuit and executing the first one or more instruction sets for operating the avatar of the application

correlated with the first collection of object representations following the interrupt. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes causing the application to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying one or more instruction sets of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes redirecting the application to the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes redirecting the application to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes causing the avatar of the application to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying one or more instruction sets of the avatar of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying the avatar of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes redirecting the avatar of the application to the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes redirecting the avatar of the application to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying at least one of: the memory unit, a register of the processor circuit, a storage, or a repository where instruction sets are stored or used.

In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying at least one of: an element of the processor circuit, an element of the application, an element of the avatar of the application, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of the avatar of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes utilizing an assembly language. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes adding or inserting additional code into a code of the application. In further embodiments, the causing the processor circuit

to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes at least one of: modifying, removing, rewriting, or overwriting a code of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes adding or inserting additional code into a code of the avatar of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes at least one of: modifying, removing, rewriting, or overwriting a code of the avatar of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of the avatar of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes implementing a user's knowledge, style, or methodology of operating the avatar of the application in a circumstance. In further embodiments, the system further comprises: an interface configured to cause execution of instruction sets, wherein the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations by the processor circuit is caused by the interface. The interface may include a modification interface.

In some embodiments, the avatar's performing the one or more operations defined by the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes implementing a user's knowledge, style, or methodology of operating the avatar of the application in a circumstance.

In certain embodiments, the artificial intelligence unit is further configured to: receive at least one extra information. In further embodiments, the at least one extra information include one or more of: a time information, a location information, a computed information, a visual information, an acoustic information, or a contextual information. In further embodiments, the at least one extra information include one or more of: an information on the avatar of the application, an information on the avatar's circumstance, an information on an object, an information on an object representation, an information on a collection of object representations, an information on an instruction set, an information on the application, an information on the processor circuit, or an information on a user. In further embodiments, the artificial intelligence unit is further configured to: learn the first collection of object representations correlated with the at least one extra information. The learning the first collection of object representations correlated with at least one extra information may include correlating the first collection of object representations with the at least one extra information. The learning the first collection of object representations correlated with at least one extra information may include storing the first collection of object representations correlated with the at least one extra information into the memory unit. In further embodiments, the anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object

representations includes anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations based on at least a partial match between an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations. The anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations based on at least a partial match between an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations may include comparing an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations. The anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations based on at least a partial match between an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations may include determining that a similarity between an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations exceeds a similarity threshold.

In some embodiments, the system of further comprises: a user interface, wherein the artificial intelligence unit is further configured to: cause the user interface to present a user with an option to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the system of further comprises: a user interface, wherein the artificial intelligence unit is further configured to: receive, via the user interface, a user's selection to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations.

In further embodiments, the artificial intelligence unit is further configured to: rate the executed first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. The rating the executed first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations may include causing a user interface to display the executed first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations along with one or more rating values as options to be selected by a user. The rating the executed first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations may include rating the executed first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations without a user input.

In certain embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: cause the user interface to present a user with an option to cancel the execution of the executed first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the canceling the execution of the executed first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes restoring the processor circuit, the application, or the avatar of the application to a prior state. The restoring the processor circuit, the application, or the avatar of the application to a prior state may include saving the state of the processor circuit, the application, or the avatar of the application prior to executing the first

one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations.

In some embodiments, the system further comprises: an input device configured to receive a user's operating directions, the user's operating directions for instructing the processor circuit, the application, or the avatar of the application on how to operate the avatar of the application.

In certain embodiments, the autonomous avatar operating includes a partially or a fully autonomous avatar operating. The partially autonomous avatar operating may include executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations responsive to a user confirmation. The fully autonomous avatar operating may include executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations without a user confirmation.

In some embodiments, the artificial intelligence unit is further configured to: receive a second collection of object representations, the second collection of object representations including one or more object representations representing one or more objects of the application; receive a second one or more instruction sets for operating an avatar of the application; and learn the second collection of object representations correlated with the second one or more instruction sets for operating the avatar of the application. In further embodiments, the second collection of object representations is received at a second time. In further embodiments, the second collection of object representations includes a unit of knowledge of the avatar's circumstance at a second time. In further embodiments, the second collection of object representations includes a stream of collections of object representations. In further embodiments, the second collection of object representations includes or is associated with a time stamp, an order, or a time related information. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application and the learning the second collection of object representations correlated with the second one or more instruction sets for operating the avatar of the application include creating a connection between the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application and the second collection of object representations correlated with the second one or more instruction sets for operating the avatar of the application. The connection may include or be associated with at least one of: an occurrence count, a weight, a parameter, or a data. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application and the learning the second collection of object representations correlated with the second one or more instruction sets for operating the avatar of the application include updating a connection between the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application and the second collection of object representations correlated with the second one or more instruction sets for operating the avatar of the application. The updating the connection between the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application and the second collection of object representations correlated with the second one or more instruction sets for operating the avatar of the application may include updating at least one of: an occurrence count, a weight, a parameter, or a data included in or associated with the connection. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application includes storing the first collection of object

representations correlated with the first one or more instruction sets for operating the avatar of the application into a first node of a knowledgebase, and wherein the learning the second collection of object representations correlated with the second one or more instruction sets for operating the avatar of the application includes storing the second collection of object representations correlated with the second one or more instruction sets for operating the avatar of the application into a second node of the knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. The knowledgebase may be stored in the memory unit. The learning the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application and the learning the second collection of object representations correlated with the second one or more instruction sets for operating the avatar of the application may include creating a connection between the first node and the second node. The learning the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application and the learning the second collection of object representations correlated with the second one or more instruction sets for operating the avatar of the application may include updating a connection between the first node and the second node. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application is stored into a first node of a neural network and the second collection of object representations correlated with the second one or more instruction sets for operating the avatar of the application is stored into a second node of the neural network. The first node and the second node may be connected by a connection. The first node may be part of a first layer of the neural network and the second node may be part of a second layer of the neural network. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application is stored into a first node of a graph and the second collection of object representations correlated with the second one or more instruction sets for operating the avatar of the application is stored into a second node of the graph. The first node and the second node may be connected by a connection. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application is stored into a first node of a sequence and the second collection of object representations correlated with the second one or more instruction sets for operating the avatar of the application is stored into a second node of the sequence.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first collection of object representations, the first collection of object representations including one or more object representations representing one or more objects of an application. The operations may further comprise: receiving a first one or more instruction sets for operating an avatar of the application. The operations may further comprise: learning the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application. The operations may further comprise: receiving a new collection of object representations, the new collection of object representations including one or more object representations representing one or more objects of the application. The operations may further comprise: anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations.

The operations may further comprise: causing an execution of the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, wherein the avatar of the application performs one or more operations defined by the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations.

In some embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes receiving the first one or more instruction sets for operating the avatar of the application from the one or more processor circuits or from another one or more processor circuits. In further embodiments, the execution of the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations is performed by the one or more processor circuits or by another one or more processor circuits.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a processor circuit, the first collection of object representations including one or more object representations representing one or more objects of an application. The method may further comprise: (b) receiving a first one or more instruction sets for operating an avatar of the application by the processor circuit. The method may further comprise: (c) learning the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application, the learning of (c) performed by the processor circuit. The method may further comprise: (d) receiving a new collection of object representations by the processor circuit, the new collection of object representations including one or more object representations representing one or more objects of the application. The method may further comprise: (e) anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (e) performed by the processor circuit. The method may further comprise: (f) executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations, the executing of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the avatar of the application, one or more operations defined by the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations.

In certain embodiments, the receiving of (b) includes receiving the first one or more instruction sets for operating the avatar of the application from the processor circuit or from another processor circuit. In further embodiments, the executing of (f) is performed by the processor circuit or by another processor circuit.

The aforementioned system, the non-transitory computer storage medium, and/or the method may include any elements, operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable as well as the following embodiments.

In some embodiments, the first one or more instruction sets for operating the avatar of the application include one or more instruction sets executed in operating the avatar of the application. In further embodiments, the first one or more instruction sets for operating the avatar of the application are part of the application. In further embodiments, the first one or more instruction sets for operating the avatar of the application are part of the avatar

of the application. In further embodiments, the first one or more instruction sets for operating the avatar of the application include one or more inputs into or one or more outputs from a processor circuit. In further embodiments, the first one or more instruction sets for operating the avatar of the application include a value or a state of a register or an element of a processor circuit. In further embodiments, the first one or more instruction sets for operating the avatar of the application include at least one of: a command, a keyword, a symbol, an instruction, an operator, a variable, a value, an object, a data structure, a function, a parameter, a state, a signal, an input, an output, a character, a digit, or a reference thereto. In further embodiments, the first one or more instruction sets for operating the avatar of the application include a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, a structured query language (SQL) code, or a machine code. In further embodiments, the first one or more instruction sets for operating the avatar of the application include one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the first one or more instruction sets for operating the avatar of the application include one or more instruction sets for operating the application.

In certain embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes receiving the first one or more instruction sets for operating the avatar of the application executed by a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes receiving the first one or more instruction sets for operating the avatar of the application as they are executed by a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes obtaining the first one or more instruction sets for operating the avatar of the application from a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes receiving the first one or more instruction sets for operating the avatar of the application from a register or an element of a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes receiving the first one or more instruction sets for operating the avatar of the application from at least one of: the memory unit, a virtual machine, a runtime engine, a hard drive, a storage device, a peripheral device, a network connected device, or a user. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes receiving the first one or more instruction sets for operating the avatar of the application from a plurality of processor circuits, applications, memory units, virtual machines, runtime engines, hard drives, storage devices, peripheral devices, network connected devices, or users. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes obtaining the first one or more instruction sets for operating the avatar of the application from the application. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes obtaining the first one or more instruction sets for operating the avatar of the application from the avatar of the application. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes receiving the one or more instruction sets for operating the avatar of the application at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: tracing, profiling, or instrumentation of a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, a

structured query language (SQL) code, or a machine code. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: tracing, profiling, or instrumentation of a register of a processor circuit, a memory unit, a storage, or a repository where the first one or more instruction sets for operating the avatar of the application are stored. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: tracing, profiling, or instrumentation of a processor circuit, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a processing element. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: tracing, profiling, or instrumentation of a processor circuit or tracing, profiling, or instrumentation of a component of a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: tracing, profiling, or instrumentation of the application. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: tracing, profiling, or instrumentation of the avatar of the application. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: tracing, profiling, or instrumentation at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: tracing, profiling, or instrumentation of one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: tracing, profiling, or instrumentation of a user input. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes at least one of: a manual, an automatic, a dynamic, or a just in time (JIT) tracing, profiling, or instrumentation. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes utilizing at least one of: a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, a logging tool, or an independent tool for obtaining instruction sets. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes utilizing an assembly language. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes utilizing a branch or a jump. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes a branch tracing or a simulation tracing. In further embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes receiving the first one or more instruction sets for operating the avatar of the application via an interface. The interface may include an acquisition interface.

In some embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application includes storing the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application into a memory unit, the memory unit comprising a plurality of collections of object representations correlated with one or more instruction sets for operating the avatar of the application.

In certain embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations

instead of or prior to an instruction set that would have been executed next. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying one or more instruction sets of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying a register or an element of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes inserting the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations into a register or an element of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes redirecting a processor circuit to the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes redirecting a processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes transmitting, to a processor circuit for execution, the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes issuing an interrupt to a processor circuit and executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations following the interrupt. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes causing the application to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying one or more instruction sets of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes redirecting the application to the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes redirecting the application to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes causing the avatar of the application to execute the first one or more instruction sets for operating the avatar of the

application correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying one or more instruction sets of the avatar of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying the avatar of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes redirecting the avatar of the application to the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes redirecting the avatar of the application to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying at least one of: a memory unit, a register of a processor circuit, a storage, or a repository where instruction sets are stored or used. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying at least one of: an element of a processor circuit, an element of the application, an element of the avatar of the application, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes modifying one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of the avatar of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the executing

the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes utilizing an assembly language. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes adding or inserting additional code into a code of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes at least one of: modifying, removing, rewriting, or overwriting a code of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes adding or inserting additional code into a code of the avatar of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes at least one of: modifying, removing, rewriting, or overwriting a code of the avatar of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of the avatar of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes implementing a user's knowledge, style, or methodology of operating the avatar of the application in a circumstance. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations includes executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations via an interface. The interface may include a modification interface.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving at least one extra information. In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: learning the first collection of object representations correlated with the at least one extra information. In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: presenting, via a user interface, a user with an option

to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving, via a user interface, a user's selection to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: rating the executed first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: presenting, via a user interface, a user with an option to cancel the execution of the executed first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving, via an input device, a user's operating directions, the user's operating directions for instructing the processor circuit, the application, or the avatar of the application on how to operate the avatar of the application. In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving a second collection of object representations, the second collection of object representations including one or more object representations representing one or more objects of the application; receiving a second one or more instruction sets for operating the avatar of the application; and learning the second collection of object representations correlated with the second one or more instruction sets for operating the avatar of the application.

In some aspects, the disclosure relates to a system for learning an avatar's circumstances for autonomous avatar operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets of an application. The system may further comprise: a memory unit configured to store data. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first collection of object representations, the first collection of object representations including one or more object representations representing one or more objects of the application. The artificial intelligence unit may be further configured to: receive a first one or more instruction sets for operating an avatar of the application. The artificial intelligence unit may be further configured to: learn the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first collection of object representations, the first collection of object representations including one or more object representations representing one or more objects of an application. The operations may further comprise: receiving a first one or more instruction sets for operating an avatar of the application. The operations may further comprise: learning the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a processor circuit, the first collection of object representations including one or more object representations representing one or more objects of an application. The method may further comprise: (b) receiving

a first one or more instruction sets for operating an avatar of the application by the processor circuit. The method may further comprise: (c) learning the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application, the learning of (c) performed by the processor circuit.

The aforementioned system, the non-transitory computer storage medium, and/or the method may include any elements, operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable as well as the following embodiments.

In some aspects, the disclosure relates to a system for using an avatar's circumstances for autonomous avatar operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets of an application. The system may further comprise: a memory unit configured to store data. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: access the memory unit that comprises a plurality of collections of object representations correlated with one or more instruction sets for operating an avatar of the application, the plurality of collections of object representations correlated with one or more instruction sets for operating the avatar of the application including a first collection of object representations correlated with a first one or more instruction sets for operating the avatar of the application. The artificial intelligence unit may be further configured to: receive a new collection of object representations, the new collection of object representations including one or more object representations representing one or more objects of the application. The artificial intelligence unit may be further configured to: anticipate the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the avatar of the application performs one or more operations defined by the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: accessing a memory unit that comprises a plurality of collections of object representations correlated with one or more instruction sets for operating an avatar of an application, the plurality of collections of object representations correlated with one or more instruction sets for operating the avatar of the application including a first collection of object representations correlated with a first one or more instruction sets for operating the avatar of the application. The operations may further comprise: receiving a new collection of object representations, the new collection of object representations including one or more object representations representing one or more objects of the application. The operations may further comprise: anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further comprise: causing an execution of the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations, the causing performed in response to the anticipating the first one or more

instruction sets for operating the avatar of the application correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, wherein the avatar of the application performs one or more operations defined by the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations.

In some aspects, the disclosure relates to a method comprising: (a) accessing a memory unit that comprises a plurality of collections of object representations correlated with one or more instruction sets for operating an avatar of an application, the plurality of collections of object representations correlated with one or more instruction sets for operating the avatar of the application including a first collection of object representations correlated with a first one or more instruction sets for operating the avatar of the application, the accessing of (a) performed by a processor circuit. The method may further comprise: (b) receiving a new collection of object representations by the processor circuit, the new collection of object representations including one or more object representations representing one or more objects of the application. The method may further comprise: (c) anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (c) performed by the processor circuit. The method may further comprise: (d) executing the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations, the executing of (d) performed in response to the anticipating of (c). The method may further comprise: (e) performing, by the avatar of the application, one or more operations defined by the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations.

The aforementioned system, the non-transitory computer storage medium, and/or the method may include any elements, operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable as well as the following embodiments.

In some aspects, the disclosure relates to a system for learning and using an avatar's circumstances for autonomous avatar operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets of an application. The system may further comprise: a memory unit configured to store data. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first stream of collections of object representations, the first stream of collections of object representations including one or more object representations representing one or more objects of the application. The artificial intelligence unit may be further configured to: receive a first one or more instruction sets for operating an avatar of the application. The artificial intelligence unit may be further configured to: learn the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application. The artificial intelligence unit may be further configured to: receive a new stream of collections of object representations, the new stream of collections of object representations including one or more object representations representing one or more objects of the application. The artificial intelligence unit may be further configured to: anticipate the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations

and the first stream of collections of object representations. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the avatar of the application performs one or more operations defined by the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations.

In certain embodiments, the first stream of collections of object representations includes one or more collections of object representations, and wherein each collection of object representations of the first stream of collections of object representations includes one or more object representations. In further embodiments, the new stream of collections of object representations includes one or more collections of object representations, and wherein each collection of object representations of the new stream of collections of object representations includes one or more object representations. In further embodiments, the first stream of collections of object representations is received over a first time period. In further embodiments, the new stream of collections of object representations is received over a new time period. In further embodiments, the first stream of collections of object representations includes a unit of knowledge of the avatar's circumstance over a first time period. In further embodiments, the new stream of collections of object representations includes a unit of knowledge of the avatar's circumstance over a new time period. In further embodiments, an object representation includes one or more properties of an object of the application. In further embodiments, an object representation includes one or more information on an object of the application. In further embodiments, the first or the new stream of collections of object representations includes or is associated with a time stamp, an order, or a time related information. In further embodiments, the first stream of collections of object representations includes a comparative stream of collections of object representations whose at least one portion can be used for comparisons with at least one portion of streams of collections of object representations subsequent to the first stream of collections of object representations, the streams of collections of object representations subsequent to the first stream of collections of object representations comprising the new stream of collections of object representations. In further embodiments, the first stream of collections of object representations includes a comparative stream of collections of object representations that can be used for comparison with the new stream of collections of object representations. In further embodiments, the new stream of collections of object representations includes an anticipatory stream of collections of object representations that can be compared with streams of collections of object representations whose correlated one or more instruction sets for operating the avatar of the application can be used for anticipation of one or more instruction sets to be executed in autonomous operating of the avatar of the application.

In some embodiments, the receiving the first stream of collections of object representations includes receiving one or more properties of the one or more objects of the application. The one or more properties of the one or more objects of the application may include one or more information on the one or more objects of the application. The receiving the one or more properties of the one or more objects of the application may include receiving the one or more properties of the one or more objects of the application from an engine, an environment, or a system used to implement the application. The receiving the one or more properties of the one or more objects of the application may include at least one of: accessing or reading a scene graph or a data structure used for organizing the one or more objects of the application. The receiving the one or more properties of the one or more objects of the

application may include detecting the one or more properties of the one or more objects of the application in a picture of the avatar's surrounding. The receiving the one or more properties of the one or more objects of the application may include detecting the one or more properties of the one or more objects of the application in a sound from the avatar's surrounding.

In certain embodiments, the system further comprises: an object processing unit configured to receive streams of collections of object representations, wherein the first or the new stream of collections of object representations is received by the object processing unit.

In some embodiments, the first one or more instruction sets for operating the avatar of the application include one or more instruction sets that temporally correspond to the first stream of collections of object representations. The one or more instruction sets that temporally correspond to the first stream of collections of object representations may include one or more instruction sets executed at a time of generating the first stream of collections of object representations. The one or more instruction sets that temporally correspond to the first stream of collections of object representations may include one or more instruction sets executed prior to generating the first stream of collections of object representations. The one or more instruction sets that temporally correspond to the first stream of collections of object representations may include one or more instruction sets executed within a threshold period of time prior to generating the first stream of collections of object representations. The one or more instruction sets that temporally correspond to the first stream of collections of object representations may include one or more instruction sets executed subsequent to generating the first stream of collections of object representations. The one or more instruction sets that temporally correspond to the first stream of collections of object representations may include one or more instruction sets executed within a threshold period of time subsequent to generating the first stream of collections of object representations. The one or more instruction sets that temporally correspond to the first stream of collections of object representations may include one or more instruction sets executed within a threshold period of time prior to generating the first stream of collections of object representations and a threshold period of time subsequent to generating the first stream of collections of object representations.

In some embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application include a unit of knowledge of how the avatar of the application operated in a circumstance. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application are included in a neuron, a node, a vertex, or an element of a knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. Some of the neurons, nodes, vertices, or elements are interconnected. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application are structured into a knowledge cell. In further embodiments, the knowledge cell is included in a neuron, a node, a vertex, or an element of a knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. Some of the neurons, nodes, vertices, or elements are interconnected.

In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application includes correlating the first stream of

collections of object representations with the first one or more instruction sets for operating the avatar of the application. The correlating the first stream of collections of object representations with the first one or more instruction sets for operating the avatar of the application may include generating a knowledge cell, the knowledge cell comprising the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application. The correlating the first stream of collections of object representations with the first one or more instruction sets for operating the avatar of the application may include structuring a unit of knowledge of how the avatar of the application operated in a circumstance. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application includes learning a user's knowledge, style, or methodology of operating the avatar of the application in a circumstance.

In certain embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application includes storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application into the memory unit, the memory unit comprising a plurality of streams of collections of object representations correlated with one or more instruction sets for operating the avatar of the application. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the avatar of the application include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the avatar of the application are organized into a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, one or more streams of collections of object representations correlated with one or more instruction sets for operating the avatar of the application of the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the avatar of the application are included in one or more neurons, nodes, vertices, or elements of a knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. Some of the neurons, nodes, vertices, or elements are interconnected. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the avatar of the application include a user's knowledge, style, or methodology of operating the avatar of the application in circumstances. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the avatar of the application are stored on a remote computing device or a remote computing system. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the avatar of the application include an artificial intelligence system for knowledge structuring, storing, or representation. The artificial intelligence system for knowledge structuring, storing, or representation may include at least one of: a deep learning system, a supervised learning system, an unsupervised learning system, a neural network, a search-based system, an optimization-based system, a logic-based system, a fuzzy logic-based system, a tree-based system, a graph-based system, a hierarchical system, a symbolic system, a sub-symbolic system, an evolutionary system, a genetic system, a multi-agent system, a deterministic system, a probabilistic system, or a statistical system.

In some embodiments, the anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes comparing at least one portion of the new stream of collections of object representations with at least one portion of the first stream of collections of object representations. In further embodiments, the at least one portion of the new stream of collections of object representations include at least one collection of object representations, at least one object representation, or at least one object property of the new stream of collections of object representations. In further embodiments, the at least one portion of the first stream of collections of object representations include at least one collection of object representations, at least one object representation, or at least one object property of the first stream of collections of object representations. In further embodiments, the anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes comparing at least one collection of object representations from the new stream of collections of object representations with at least one collection of object representations from the first stream of collections of object representations. In further embodiments, the comparing at least one collection of object representations from the new stream of collections of object representations with at least one collection of object representations from the first stream of collections of object representations includes comparing at least one object representation of the at least one collection of object representations from the new stream of collections of object representations with at least one object representation of the at least one collection of object representations from the first stream of collections of object representations. The comparing at least one object representation of the at least one collection of object representations from the new stream of collections of object representations with at least one object representation of the at least one collection of object representations from the first stream of collections of object representations may include comparing at least one object property of the at least one object representation of the at least one collection of object representations from the new stream of collections of object representations with at least one object property of the at least one object representation of the at least one collection of object representations from the first stream of collections of object representations. In further embodiments, the anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes comparing at least one object representation of the at least one collection of object representations from the new stream of collections of object representations with at least one object representation of the at least one collection of object representations from the first stream of collections of object representations. In further embodiments, the anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes comparing at least one object property of at least one object representation of at least one collection of object representations from the new stream of collections of object representations with at least one object property of at least one object representation of at least one collection of object representations from the first stream of collections of object representations.

In certain embodiments, the anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that there is at least a partial match between one or more portions of the new stream of collections of object representations and one or more portions of the first stream of collections of object representations. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that a similarity between at least one portion of the new stream of collections of object representations and at least one portion of the first stream of collections of object representations exceeds a similarity threshold. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining a substantial similarity between at least one portion of the new stream of collections of object representations and at least one portion of the first stream of collections of object representations. The substantial similarity may be achieved when a similarity between the at least one portion of the new stream of collections of object representations and the at least one portion of the first stream of collections of object representations exceeds a similarity threshold. The substantial similarity may be achieved when a number or a percentage of matching or partially matching portions of the new stream of collections of object representations and portions of the first stream of collections of object representations exceeds a threshold number or threshold percentage. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that a number or a percentage of matching or partially matching collections of object representations from the new stream of collections of object representations and from the first stream of collections of object representations exceeds a threshold number or threshold percentage. The matching or partially matching collections of object representations from the new stream of collections of object representations and from the first stream of collections of object representations may be determined factoring in at least one of: an importance of a collection of object representations, an order of a collection of object representations, a threshold for a similarity in a collection of object representations, or a threshold for a difference in a collection of object representations. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that a number or a percentage of matching or partially matching object representations from the new stream of collections of object representations and from the first stream of collections of object representations exceeds a threshold number or threshold percentage. The matching or partially matching object representations from the new stream of collections of object representations and from the first stream of collections of object representations may be determined factoring in at least one of: an association of an object representation with a collection of object representations, a type of an object representation, an importance of an object representation, a threshold for a similarity in an object representation, or a threshold for a difference in an object representation. In further embodiments, the determining

that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that a number or a percentage of matching or partially matching object properties from the new stream of collections of object representations and from the first stream of collections of object representations exceeds a threshold number or threshold percentage. The matching or partially matching object properties from the new stream of collections of object representations and from the first stream of collections of object representations may be determined factoring in at least one of: an association of an object property with an object representation, an association of an object property with a collection of object representations, a category of an object property, an importance of an object property, a threshold for a similarity in an object property, or a threshold for a difference in an object property. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that there is at least a partial match between at least one collection of object representations from the new stream of collections of object representations and at least one collection of object representations from the first stream of collections of object representations. The determining that there is at least a partial match between at least one collection of object representations from the new stream of collections of object representations and at least one collection of object representations from the first stream of collections of object representations may include determining that there is at least a partial match between at least one object representation of the at least one collection of object representations from the new stream of collections of object representations and at least one object representation of the at least one collection of object representations from the first stream of collections of object representations. The determining that there is at least a partial match between at least one object representation of the at least one collection of object representations from the new stream of collections of object representations and at least one object representation of the at least one collection of object representations from the first stream of collections of object representations may include determining that there is at least a partial match between at least one object property of the at least one object representation of the at least one collection of object representations from the new stream of collections of object representations and at least one object property of the at least one object representation of the at least one collection of object representations from the first stream of collections of object representations.

In some embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations instead of or prior to an instruction set that would have been executed next. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying one or more instruction sets of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes inserting the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations into a register or

an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes redirecting the processor circuit to the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes redirecting the processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes transmitting, to the processor circuit for execution, the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes issuing an interrupt to the processor circuit and executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations following the interrupt. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes causing the application to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying one or more instruction sets of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes redirecting the application to the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes redirecting the application to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes causing the avatar of the application to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying one or more instruction sets of the avatar of the application. In further embodiments, the causing

the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying the avatar of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes redirecting the avatar of the application to the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes redirecting the avatar of the application to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying at least one of: the memory unit, a register of the processor circuit, a storage, or a repository where instruction sets are stored or used. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying at least one of: an element of the processor circuit, an element of the application, an element of the avatar of the application, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of the avatar of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of

object representations includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes utilizing an assembly language. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes adding or inserting additional code into a code of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes at least one of: modifying, removing, rewriting, or overwriting a code of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes adding or inserting additional code into a code of the avatar of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes at least one of: modifying, removing, rewriting, or overwriting a code of the avatar of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of the avatar of the application. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes implementing a user's knowledge, style, or methodology of operating the avatar of the application in a circumstance. In further embodiments, the system further comprises: an interface configured to cause execution of instruction sets, wherein the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations by the processor circuit is caused by the interface. The interface may include a modification interface.

In certain embodiments, the avatar's performing the one or more operations defined by the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes implementing a user's knowledge, style, or methodology of operating the avatar of the application in a circumstance.

In some embodiments, the artificial intelligence unit is further configured to: receive at least one extra information. In further embodiments, the at least one extra information include one or more of: a time information, a location information, a computed information, a visual information, an acoustic information, or a contextual information. In further embodiments, the at least one extra information include one or more of: an information on the avatar of the application, an information on the avatar's circumstance, an information on an object, an information on an object representation, an information on a collection of object representations, an information on a stream of collections of object representations, an information on an instruction set, an information on the application, an information on the processor circuit, or an information on a user. In further embodiments, the artificial intelligence unit is further configured to: learn the first stream of collections of object representations correlated with the at least one extra information. The learning the first stream of collections of object representations correlated with at least one extra information may include correlating the first stream of collections of object representations with the at least one extra information. The learning the first stream of collections of object representations correlated with at least one extra information may include storing the first stream of collections of object representations correlated with the at least one extra information into the memory unit. In further embodiments, the anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based on at least a partial match between an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations. The anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based on at least a partial match between an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations may include comparing an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations. The anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based on at least a partial match between an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations may include determining that a similarity between an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations exceeds a similarity threshold.

In certain embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: cause the user interface to present a user with an option to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object

representations. In further embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: receive, via the user interface, a user's selection to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the artificial intelligence unit is further configured to: rate the executed first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. The rating the executed first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations may include causing a user interface to display the executed first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations along with one or more rating values as options to be selected by a user. The rating the executed first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations may include rating the executed first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations without a user input. In further embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: cause the user interface to present a user with an option to cancel the execution of the executed first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the canceling the execution of the executed first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes restoring the processor circuit, the application, or the avatar of the application to a prior state. The restoring the processor circuit, the application, or the avatar of the application to a prior state may include saving the state of the processor circuit, the application, or the avatar of the application prior to executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations.

In some embodiments, the system further comprises: an input device configured to receive a user's operating directions, the user's operating directions for instructing the processor circuit, the application, or the avatar of the application on how to operate the avatar of the application.

In certain embodiments, the autonomous avatar operating includes a partially or a fully autonomous avatar operating. The partially autonomous avatar operating may include executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations responsive to a user confirmation. The fully autonomous avatar operating may include executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations without a user confirmation.

In some embodiments, the artificial intelligence unit is further configured to: receive a second stream of collections of object representations, the second stream of collections of object representations including one or more object representations representing one or more objects of the application; receive a second one or more instruction sets for operating the avatar of the application; and learn the second stream of collections of object representations correlated with the second one or more instruction sets for operating the avatar of the application. In further embodiments, the second stream of collections of object representations includes one or more collections of object representations, and wherein each collection of object representations of the second stream of collections of object representations includes one or more object representations. In further embodiments, the second stream of

collections of object representations is received over a second time period. In further embodiments, the second stream of collections of object representations includes a unit of knowledge of the avatar's circumstance over a second time period. In further embodiments, the second stream of collections of object representations includes or is associated with a time stamp, an order, or a time related information. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application and the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the avatar of the application include creating a connection between the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the avatar of the application. The connection may include or is associated with at least one of: an occurrence count, a weight, a parameter, or a data. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application and the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the avatar of the application include updating a connection between the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the avatar of the application. In further embodiments, the updating the connection between the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the avatar of the application includes updating at least one of: an occurrence count, a weight, a parameter, or a data included in or associated with the connection. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application includes storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application into a first node of a knowledgebase, and wherein the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the avatar of the application includes storing the second stream of collections of object representations correlated with the second one or more instruction sets for operating the avatar of the application into a second node of the knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. The knowledgebase may be stored in the memory unit. The learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application and the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the avatar of the application may include creating a connection between the first node and the second node. The learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application and the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the avatar of the application may include updating a connection between the first node and the second node. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction

sets for operating the avatar of the application is stored into a first node of a neural network and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the avatar of the application is stored into a second node of the neural network. The first node and the second node may be connected by a connection. The first node may be part of a first layer of the neural network and the second node may be part of a second layer of the neural network. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application is stored into a first node of a graph and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the avatar of the application is stored into a second node of the graph. The first node and the second node may be connected by a connection. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application is stored into a first node of a sequence and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the avatar of the application is stored into a second node of the sequence.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first stream of collections of object representations, the first stream of collections of object representations including one or more object representations representing one or more objects of an application. The operations may further comprise: receiving a first one or more instruction sets for operating an avatar of the application. The operations may further comprise: learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application. The operations may further comprise: receiving a new stream of collections of object representations, the new stream of collections of object representations including one or more object representations representing one or more objects of the application. The operations may further comprise: anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The operations may further comprise: causing an execution of the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, wherein the avatar of the application performs one or more operations defined by the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations.

In certain embodiments, the receiving the first one or more instruction sets for operating the avatar of the application includes receiving the first one or more instruction sets for operating the avatar of the application from the one or more processor circuits or from another one or more processor circuits. In further embodiments, the execution of the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations is performed by the one or more processor circuits or by another one or more processor circuits.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first stream of collections of object representations by a processor circuit, the first stream of collections of object representations including one or more object representations representing one or more objects of an application. The method may further comprise: (b) receiving a first one or more instruction sets for operating an avatar of the application by the processor circuit. The method may further comprise: (c) learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application, the learning of (c) performed by the processor circuit. The method may further comprise: (d) receiving a new stream of collections of object representations by the processor circuit, the new stream of collections of object representations including one or more object representations representing one or more objects of the application. The method may further comprise: (e) anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, the anticipating of (e) performed by the processor circuit. The method may further comprise: (f) executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations, the executing of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the avatar of the application, one or more operations defined by the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations.

In some embodiments, the receiving of (b) includes receiving the first one or more instruction sets for operating the avatar of the application from the processor circuit or from another processor circuit. In further embodiments, the executing of (f) is performed by the processor circuit or by another processor circuit.

The aforementioned system, the non-transitory computer storage medium, and/or the method may include any elements, operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable as well as the following embodiments.

In certain embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application includes storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application into the memory unit, the memory unit comprising a plurality of streams of collections of object representations correlated with one or more instruction sets for operating the avatar of the application.

In some embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations instead of or prior to an instruction set that would have been executed next. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying one or more instruction sets of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying a register or an element of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes inserting the

first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations into a register or an element of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes redirecting a processor circuit to the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes redirecting a processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes transmitting, to a processor circuit for execution, the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes issuing an interrupt to a processor circuit and executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations following the interrupt. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes causing the application to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying one or more instruction sets of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes redirecting the application to the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes redirecting the application to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes causing the avatar of the application to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying one or more instruction sets of the avatar of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying the avatar

of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes redirecting the avatar of the application to the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes redirecting the avatar of the application to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying at least one of: a memory unit, a register of a processor circuit, a storage, or a repository where instruction sets are stored or used. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying at least one of: an element of a processor circuit, an element of the application, an element of the avatar of the application, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes modifying one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of the avatar of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations

includes utilizing an assembly language. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes adding or inserting additional code into a code of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes at least one of: modifying, removing, rewriting, or overwriting a code of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes adding or inserting additional code into a code of the avatar of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes at least one of: modifying, removing, rewriting, or overwriting a code of the avatar of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of the avatar of the application. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes implementing a user's knowledge, style, or methodology of operating the avatar of the application in a circumstance. In further embodiments, the executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations includes executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations via an interface. The interface may include a modification interface.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving at least one extra information. In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: learning the first stream of collections of object representations correlated with the at least one extra information. In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: presenting, via a user interface, a user with an option to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving, via a user interface, a user's selection to

execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: rating the executed first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: presenting, via a user interface, a user with an option to cancel the execution of the executed first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations. In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving, via an input device, a user's operating directions, the user's operating directions for instructing the processor circuit, the application, or the avatar of the application on how to operate the avatar of the application. In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving a second stream of collections of object representations, the second stream of collections of object representations including one or more object representations representing one or more objects of the application; receiving a second one or more instruction sets for operating the avatar of the application; and learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the avatar of the application.

The aforementioned system, the non-transitory computer storage medium, and/or the method may include any elements, operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable as well as the following embodiments.

In some aspects, the disclosure relates to a system for learning an avatar's circumstances for autonomous avatar operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets of an application. The system may further comprise: a memory unit configured to store data. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first stream of collections of object representations, the first stream of collections of object representations including one or more object representations representing one or more objects of the application. The artificial intelligence unit may be further configured to: receive a first one or more instruction sets for operating an avatar of the application. The artificial intelligence unit may be further configured to: learn the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first stream of collections of object representations, the first stream of collections of object representations including one or more object representations representing one or more objects of an application. The operations may further comprise: receiving a first one or more instruction sets for operating an avatar of the application. The operations may further comprise: learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first stream of collections of object representations by a processor circuit, the first stream of collections of object representations including one or

more object representations representing one or more objects of an application. The method may further comprise: (b) receiving a first one or more instruction sets for operating an avatar of the application by the processor circuit. The method may further comprise: (c) learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar of the application, the learning of (c) performed by the processor circuit.

The aforementioned system, the non-transitory computer storage medium, and/or the method may include any elements, operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable as well as the following embodiments.

In some aspects, the disclosure relates to a system for using an avatar's circumstances for autonomous avatar operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets of an application. The system may further comprise: a memory unit configured to store data. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: access the memory unit that comprises a plurality of streams of collections of object representations correlated with one or more instruction sets for operating an avatar of the application, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the avatar of the application including a first stream of collections of object representations correlated with a first one or more instruction sets for operating the avatar of the application. The artificial intelligence unit may be further configured to: receive a new stream of collections of object representations, the new stream of collections of object representations including one or more object representations representing one or more objects of the application. The artificial intelligence unit may be further configured to: anticipate the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the avatar of the application performs one or more operations defined by the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: accessing a memory unit that comprises a plurality of streams of collections of object representations correlated with one or more instruction sets for operating an avatar of an application, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the avatar of the application including a first stream of collections of object representations correlated with a first one or more instruction sets for operating the avatar of the application. The operations may further comprise: receiving a new stream of collections of object representations, the new stream of collections of object representations including one or more object representations representing one or more objects of the application. The operations may further comprise: anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based

on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The operations may further comprise: causing an execution of the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, wherein the avatar of the application performs one or more operations defined by the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations.

In some aspects, the disclosure relates to a method comprising: (a) accessing a memory unit that comprises a plurality of streams of collections of object representations correlated with one or more instruction sets for operating an avatar of an application, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the avatar of the application including a first stream of collections of object representations correlated with a first one or more instruction sets for operating the avatar of the application, the accessing of (a) performed by a processor circuit. The method may further comprise: (b) receiving a new stream of collections of object representations by the processor circuit, the new stream of collections of object representations including one or more object representations representing one or more objects of the application. The method may further comprise: (c) anticipating the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, the anticipating of (c) performed by the processor circuit. The method may further comprise: (d) executing the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations, the executing of (d) performed in response to the anticipating of (c). The method may further comprise: (e) performing, by the avatar of the application, one or more operations defined by the first one or more instruction sets for operating the avatar of the application correlated with the first stream of collections of object representations.

The aforementioned system, the non-transitory computer storage medium, and/or the method may include any elements, operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable as well as the following embodiments.

In some aspects, the disclosure relates to a system for learning and using an application's circumstances for autonomous application operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets of an application. The system may further comprise: a memory unit configured to store data. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first collection of object representations, the first collection of object representations including one or more object representations representing one or more objects of the application. The artificial intelligence unit may be further configured to: receive a first one or more instruction sets for operating the application. The artificial intelligence unit may be further configured to: learn the first collection of object representations correlated with the first one or more instruction sets for operating the application. The artificial intelligence unit may be further configured to: receive a new collection of

object representations, the new collection of object representations including one or more object representations representing one or more objects of the application. The artificial intelligence unit may be further configured to: anticipate the first one or more instruction sets for operating the application correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the first one or more instruction sets for operating the application correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the application performs one or more operations defined by the first one or more instruction sets for operating the application correlated with the first collection of object representations.

In certain embodiments, the first collection of object representations includes a unit of knowledge of the application's circumstance at a first time. In further embodiments, an application's circumstance includes one or more objects of the application.

In some embodiments, the first one or more instruction sets for operating the application include one or more instruction sets executed in operating the application. In further embodiments, the receiving the first one or more instruction sets for operating the application includes at least one of: tracing, profiling, or instrumentation of the application.

In certain embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the application include a unit of knowledge of how the application operated in a circumstance. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the application includes learning a user's knowledge, style, or methodology of operating the application in a circumstance.

In some embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the application includes storing the first collection of object representations correlated with the first one or more instruction sets for operating the application into the memory unit, the memory unit comprising a plurality of collections of object representations correlated with one or more instruction sets for operating the application. The plurality of collections of object representations correlated with one or more instruction sets for operating the application may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure.

In certain embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the application correlated with the first collection of object representations includes causing the application to execute the first one or more instruction sets for operating the application correlated with the first collection of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the application correlated with the first collection of object representations includes implementing a user's knowledge, style, or methodology of operating the application in a circumstance.

In some embodiments, the artificial intelligence unit is further configured to: receive a second collection of object representations, the second collection of object representations including one or more object representations representing one or more objects of the application; receive a second one or more instruction sets for operating the application; and learn the second collection of object representations correlated with the second one or more instruction sets for operating the application. In further embodiments, the learning the first collection of object

representations correlated with the first one or more instruction sets for operating the application includes storing the first collection of object representations correlated with the first one or more instruction sets for operating the application into a first node of a knowledgebase, and wherein the learning the second collection of object representations correlated with the second one or more instruction sets for operating the application includes storing the second collection of object representations correlated with the second one or more instruction sets for operating the application into a second node of the knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first collection of object representations, the first collection of object representations including one or more object representations representing one or more objects of an application. The operations may further comprise: receiving a first one or more instruction sets for operating the application. The operations may further comprise: learning the first collection of object representations correlated with the first one or more instruction sets for operating the application. The operations may further comprise: receiving a new collection of object representations, the new collection of object representations including one or more object representations representing one or more objects of the application. The operations may further comprise: anticipating the first one or more instruction sets for operating the application correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further comprise: causing an execution of the first one or more instruction sets for operating the application correlated with the first collection of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the application correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, wherein the application performs one or more operations defined by the first one or more instruction sets for operating the application correlated with the first collection of object representations.

In some embodiments, the receiving the first one or more instruction sets for operating the application includes receiving the first one or more instruction sets for operating the application from the one or more processor circuits or from another one or more processor circuits. In further embodiments, the execution of the first one or more instruction sets for operating the application correlated with the first collection of object representations is performed by the one or more processor circuits or by another one or more processor circuits.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a processor circuit, the first collection of object representations including one or more object representations representing one or more objects of an application. The method may further comprise: (b) receiving a first one or more instruction sets for operating an application by the processor circuit. The method may further comprise: (c) learning the first collection of object representations correlated with the first one or more instruction sets for operating the application, the learning of (c) performed by the processor circuit. The method may further comprise: (d) receiving a new collection of object representations by the processor circuit, the new collection of object representations including one or more object representations representing one or more objects of the

application. The method may further comprise: (e) anticipating the first one or more instruction sets for operating the application correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (e) performed by the processor circuit. The method may further comprise: (f) executing the first one or more instruction sets for operating the application correlated with the first collection of object representations, the executing of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the application, one or more operations defined by the first one or more instruction sets for operating the application correlated with the first collection of object representations.

In certain embodiments, the receiving of (b) includes receiving the first one or more instruction sets for operating the application from the processor circuit or from another processor circuit. In further embodiments, the executing of (f) is performed by the processor circuit or by another processor circuit.

The aforementioned system, the non-transitory computer storage medium, and/or the method may include any elements, operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable as well as the following embodiments.

In some aspects, the disclosure relates to a system for learning and using an application's circumstances for autonomous application operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets of an application. The system may further comprise: a memory unit configured to store data. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first stream of collections of object representations, the first stream of collections of object representations including one or more object representations representing one or more objects of the application. The artificial intelligence unit may be further configured to: receive a first one or more instruction sets for operating the application. The artificial intelligence unit may be further configured to: learn the first stream of collections of object representations correlated with the first one or more instruction sets for operating the application. The artificial intelligence unit may be further configured to: receive a new stream of collections of object representations, the new stream of collections of object representations including one or more object representations representing one or more objects of the application. The artificial intelligence unit may be further configured to: anticipate the first one or more instruction sets for operating the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the first one or more instruction sets for operating the application correlated with the first stream of collections of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the application performs one or more operations defined by the first one or more instruction sets for operating the application correlated with the first stream of collections of object representations.

In some embodiments, the first stream of collections of object representations includes a unit of knowledge of the application's circumstance over a first time period. In further embodiments, an application's circumstance includes one or more objects of the application.

In certain embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the application include a unit of knowledge of how the application operated in a

circumstance. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the application includes learning a user's knowledge, style, or methodology of operating the application in a circumstance.

In some embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the application includes storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the application into the memory unit, the memory unit comprising a plurality of streams of collections of object representations correlated with one or more instruction sets for operating the application. The plurality of streams of collections of object representations correlated with one or more instruction sets for operating the application may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure.

In certain embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the application correlated with the first stream of collections of object representations includes causing the application to execute the first one or more instruction sets for operating the application correlated with the first stream of collections of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the application correlated with the first stream of collections of object representations includes implementing a user's knowledge, style, or methodology of operating the application in a circumstance.

In some embodiments, the artificial intelligence unit is further configured to: receive a second stream of collections of object representations, the second stream of collections of object representations including one or more object representations representing one or more objects of the application; receive a second one or more instruction sets for operating the application; and learn the second stream of collections of object representations correlated with the second one or more instruction sets for operating the application. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the application includes storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the application into a first node of a knowledgebase, and wherein the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the application includes storing the second stream of collections of object representations correlated with the second one or more instruction sets for operating the application into a second node of the knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first stream of collections of object representations, the first stream of collections of object representations including one or more object representations representing one or more objects of an application. The operations may further comprise: receiving a first one or more instruction sets for operating the application. The operations may further comprise: learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the application. The operations may further comprise: receiving a new stream of collections of object

representations, the new stream of collections of object representations including one or more object representations representing one or more objects of the application. The operations may further comprise: anticipating the first one or more instruction sets for operating the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The operations may further comprise: causing an execution of the first one or more instruction sets for operating the application correlated with the first stream of collections of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, wherein the application performs one or more operations defined by the first one or more instruction sets for operating the application correlated with the first stream of collections of object representations.

In certain embodiments, the receiving the first one or more instruction sets for operating the application includes receiving the first one or more instruction sets for operating the application from the one or more processor circuits or from another one or more processor circuits. In further embodiments, the execution of the first one or more instruction sets for operating the application correlated with the first stream of collections of object representations is performed by the one or more processor circuits or by another one or more processor circuits.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first stream of collections of object representations by a processor circuit, the first stream of collections of object representations including one or more object representations representing one or more objects of an application. The method may further comprise: (b) receiving a first one or more instruction sets for operating the application by the processor circuit. The method may further comprise: (c) learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the application, the learning of (c) performed by the processor circuit. The method may further comprise: (d) receiving a new stream of collections of object representations by the processor circuit, the new stream of collections of object representations including one or more object representations representing one or more objects of the application. The method may further comprise: (e) anticipating the first one or more instruction sets for operating the application correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, the anticipating of (e) performed by the processor circuit. The method may further comprise: (f) executing the first one or more instruction sets for operating the application correlated with the first stream of collections of object representations, the executing of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the application, one or more operations defined by the first one or more instruction sets for operating the application correlated with the first stream of collections of object representations.

In some embodiments, the receiving of (b) includes receiving the first one or more instruction sets for operating the application from the processor circuit or from another processor circuit. In further embodiments, the executing of (f) is performed by the processor circuit or by another processor circuit.

The aforementioned system, the non-transitory computer storage medium, and/or the method may include any elements, operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable as well as the following embodiments.

Other features and advantages of the disclosure will become apparent from the following description, including the claims and drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

Fig. 1 illustrates a block diagram of Computing Device 70 that can provide processing capabilities used in some of the disclosed embodiments.

Fig. 2 illustrates an embodiment of Computing Device 70 comprising Unit for Learning and/or Using an Avatar's Circumstances for Autonomous Avatar Operation (ACAAO Unit 100).

Fig. 3 illustrates an embodiment of utilizing Picture Renderer 91 and Picture Recognizer 92.

Fig. 4 illustrates an embodiment of utilizing Sound Renderer 96 and Sound Recognizer 97.

Figs. 5A-5B, illustrate an exemplary embodiment of Objects 615 in Avatar's 605 surrounding, and resulting Collection of Object Representations 525

Fig. 6 illustrates some embodiments of obtaining instruction sets, data, and/or other information through tracing, profiling, or sampling of Processor 11 registers, memory, or other computing system components.

Figs. 7A-7E illustrate some embodiments of Instruction Sets 526.

Figs. 8A-8B illustrate some embodiments of Extra Information 527.

Fig. 9 illustrates an embodiment where ACAAO Unit 100 is part of or operating on Processor 11.

Fig. 10 illustrates an embodiment where ACAAO Unit 100 resides on Server 96 accessible over Network 95.

Fig. 11 illustrates an embodiment of Artificial Intelligence Unit 110.

Fig. 12 illustrates an embodiment of Knowledge Structuring Unit 520 correlating individual Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527.

Fig. 13 illustrates another embodiment of Knowledge Structuring Unit 520 correlating individual Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527.

Fig. 14 illustrates an embodiment of Knowledge Structuring Unit 520 correlating streams of Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527.

Fig. 15 illustrates another embodiment of Knowledge Structuring Unit 520 correlating streams of Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527.

Fig. 16 illustrates various artificial intelligence methods, systems, and/or models that can be utilized in ACAAO Unit 100 embodiments.

Figs. 17A-17C illustrate embodiments of interconnected Knowledge Cells 800 and updating weights of Connections 853.

Fig. 18 illustrates an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Collection of Knowledge Cells 530d.

Fig. 19 illustrates an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Neural Network 530a.

Fig. 20 illustrates an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Neural Network 530a comprising shortcut Connections 853.

Fig. 21 illustrates an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Graph 530b.

Fig. 22 illustrates an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Collection of Sequences 530c.

Fig. 23 illustrates an embodiment of determining anticipatory Instruction Sets 526 from a single Knowledge Cell 800.

Fig. 24 illustrates an embodiment of determining anticipatory Instruction Sets 526 by traversing a single Knowledge Cell 800.

Fig. 25 illustrates an embodiment of determining anticipatory Instruction Sets 526 using collective similarity comparisons.

Fig. 26 illustrates an embodiment of determining anticipatory Instruction Sets 526 using Neural Network 530a.

Fig. 27 illustrates an embodiment of determining anticipatory Instruction Sets 526 using Graph 530b.

Fig. 28 illustrates an embodiment of determining anticipatory Instruction Sets 526 using Collection of Sequences 530c.

Fig. 29 illustrates some embodiments of modifying execution and/or functionality of Avatar 605 and/or Application Program 18 through modification of Processor 11 registers, memory, or other computing system components.

Fig. 30 illustrates a flow chart diagram of an embodiment of method 9100 for learning and/or using an avatar's circumstances for autonomous avatar operation.

Fig. 31 illustrates a flow chart diagram of an embodiment of method 9200 for learning and/or using an avatar's circumstances for autonomous avatar operation.

Fig. 32 illustrates a flow chart diagram of an embodiment of method 9300 for learning and/or using an avatar's circumstances for autonomous avatar operation.

Fig. 33 illustrates a flow chart diagram of an embodiment of method 9400 for learning and/or using an application's circumstances for autonomous application operation.

Fig. 34 illustrates a flow chart diagram of an embodiment of method 9500 for learning and/or using an application's circumstances for autonomous application operation.

Fig. 35 illustrates a flow chart diagram of an embodiment of method 9600 for learning and/or using an application's circumstances for autonomous application operation.

Fig. 36 illustrates an exemplary embodiment of Soldier 605a within 3D Computer Game 18a.

Fig. 37 illustrates an exemplary embodiment of Tank 605b within 2D Computer Game 18b.

Fig. 38 illustrates an exemplary embodiment of utilizing Area of Interest 450 around Tank 605b.

Fig. 39 illustrates an exemplary embodiment of multiple Avatars 605 within Computer Game 18c.

Like reference numerals in different figures indicate like elements. Horizontal or vertical "..." or other such indicia may be used to indicate additional instances of the same type of element. n, m, x, or other such letters or indicia represent integers or other sequential numbers that follow the sequence where they are indicated. It should be noted that n, m, x, or other such letters or indicia may represent different numbers in different elements even where the elements are depicted in the same figure. In general, n, m, x, or other such letters or indicia may follow the sequence and/or context where they are indicated. Any of these or other such letters or indicia may be used interchangeably depending on context and space available. The drawings are not necessarily to scale, with emphasis instead being placed upon illustrating the embodiments, principles, and concepts of the disclosure. A line or arrow between any of the disclosed elements comprises an interface that enables the coupling, connection, and/or interaction between the elements.

DETAILED DESCRIPTION

The disclosed artificially intelligent devices, systems, and methods for learning and/or using an avatar's circumstances for autonomous avatar operation comprise apparatuses, systems, methods, features, functionalities, and/or applications that enable learning an avatar's circumstances including objects with various properties along with correlated instruction sets for operating the avatar, storing this knowledge in a knowledgebase (i.e. neural network, graph, sequences, etc.), and/or operating an avatar autonomously. The disclosed artificially intelligent devices, systems, and methods for learning and/or using an avatar's circumstances for autonomous avatar operation, any of their elements, any of their embodiments, or a combination thereof can generally be referred to as ACAAO, ACAAO Unit, or as other suitable name or reference.

Referring now to Fig. 1, an embodiment is illustrated of Computing Device 70 (also referred to simply as computing device, computing system, or other suitable name or reference, etc.) that can provide processing capabilities used in some embodiments of the forthcoming disclosure. Later described devices, systems, and methods, in combination with processing capabilities of Computing Device 70, enable learning and/or using an avatar's circumstances for autonomous avatar operation and/or other functionalities described herein. Various embodiments of the disclosed devices, systems, and methods include hardware, functions, logic, programs, and/or a combination thereof that can be implemented using any type or form of computing, computing enabled, or other device or system such as a mobile device, a computer, a computing enabled telephone, a server, a gaming device, a television device, a digital camera, a GPS receiver, a media player, an embedded device, a supercomputer, a wearable device, an implantable device, a cloud, or any other type or form of computing, computing enabled, or other device or system capable of performing the operations described herein.

In some designs, Computing Device 70 comprises hardware, processing techniques or capabilities, programs, or a combination thereof. Computing Device 70 includes one or more central processing units, which may also be referred to as processors 11. Processor 11 includes one or more memory ports 10 and/or one or more input-output ports, also referred to as I/O ports 15, such as I/O ports 15A and 15B. Processor 11 may be special or general purpose. Computing Device 70 may further include memory 12, which can be connected to the remainder of the components of Computing Device 70 via bus 5. Memory 12 can be connected to processor 11 via memory port

10. Computing Device 70 may also include display device 21 such as a monitor, projector, glasses, and/or other display device. Computing Device 70 may also include Human-machine Interface 23 such as a keyboard, a pointing device, a mouse, a touchscreen, a joystick, a remote controller, and/or other input device. In some implementations, Human-machine Interface 23 can be connected with bus 5 or directly connected with specific elements of Computing Device 70. Computing Device 70 may include additional elements such as one or more input/output devices 13. Processor 11 may include or be interfaced with cache memory 14. Storage 27 may include memory, which provides an operating system 17 (i.e. also referred to as OS 17, etc.), additional application programs 18, and/or data space 19 in which additional data or information can be stored. Alternative memory device 16 can be connected to the remaining components of Computing Device 70 via bus 5. Network interface 25 can also be connected with bus 5 and be used to communicate with external computing devices via a network. Some or all described elements of Computing Device 70 can be directly or operatively connected or coupled with each other using any other connection means known in art. Other additional elements may be included as needed, or some of the disclosed ones may be excluded, or a combination thereof may be utilized in alternate implementations of Computing Device 70.

Processor 11 includes one or more circuits or devices that can execute instructions fetched from memory 12 and/or other element. Processor 11 may include any combination of hardware and/or processing techniques or capabilities for executing or implementing logic functions or programs. Processor 11 may include a single core or a multi core processor. Processor 11 includes the functionality for loading operating system 17 and operating any application programs 18 thereon. In some embodiments, Processor 11 can be provided in a microprocessing or a processing unit, such as, for example, Snapdragon processor produced by Qualcomm Inc., processor by Intel Corporation of Mountain View, California, processor manufactured by Motorola Corporation of Schaumburg, Ill.; processor manufactured by Transmeta Corporation of Santa Clara, Calif.; processor manufactured by International Business Machines of White Plains, N.Y.; processor manufactured by Advanced Micro Devices of Sunnyvale, California, or any computing circuit or device for performing similar functions. In other embodiments, processor 11 can be provided in a graphics processing unit (GPU), visual processing unit (VPU), or other highly parallel processing circuit or device such as, for example, nVidia GeForce line of GPUs, AMD Radeon line of GPUs, and/or others. Such GPUs or other highly parallel processing circuits or devices may provide superior performance in processing operations on neural networks, graphs, and/or other data structures. In further embodiments, processor 11 can be provided in a micro controller such as, for example, Texas instruments, Atmel, Microchip Technology, ARM, Silicon Labs, Intel, and/or other lines of micro controllers. In further embodiments, processor 11 can be provided in a quantum processor such as, for example, D-Wave Systems, Microsoft, Intel, IBM, Google, Toshiba, and/or other lines of quantum processors. In further embodiments, processor 11 can be provided in a biocomputer such as DNA-based computer, protein-based computer, molecule-based computer, and/or others. In further embodiments, processor 11 includes any circuit or device for performing logic operations. Processor 11 can be based on any of the aforementioned or other available processors capable of operating as described herein. Computing Device 70 may include one or more of the aforementioned or other processors. In some designs, processor 11 can communicate with memory 12 via a system bus 5. In other designs, processor 11 can communicate directly with memory 12 via a memory port 10.

Memory 12 includes one or more circuits or devices capable of storing data. In some embodiments, Memory 12 can be provided in a semiconductor or electronic memory chip such as static random access memory (SRAM), Flash memory, Burst SRAM or SynchBurst SRAM (BSRAM), Dynamic random access memory (DRAM), Fast Page Mode DRAM (FPM DRAM), Enhanced DRAM (EDRAM), Extended Data Output RAM (EDO RAM), Extended Data Output DRAM (EDO DRAM), Burst Extended Data Output DRAM (BEDO DRAM), Enhanced DRAM (EDRAM), synchronous DRAM (SDRAM), JEDEC SRAM, PC100 SDRAM, Double Data Rate SDRAM (DDR SDRAM), Enhanced SDRAM (ESDRAM), SyncLink DRAM (SLDRAM), Direct Rambus DRAM (DRDRAM), Ferroelectric RAM (FRAM), and/or others. In other embodiments, Memory 12 includes any volatile memory. In general, Memory 12 can be based on any of the aforementioned or other available memories capable of operating as described herein.

Storage 27 includes one or more devices or mediums capable of storing data. In some embodiments, Storage 27 can be provided in a device or medium such as a hard drive, flash drive, optical disk, and/or others. In other embodiments, Storage 27 can be provided in a biological storage device such as DNA-based storage device, protein-based storage device, molecule-based storage device, and/or others. In further embodiments, Storage 27 can be provided in an optical storage device such as holographic storage, and/or others. In further embodiments, Storage 27 may include any non-volatile memory. In general, Storage 27 can be based on any of the aforementioned or other available storage devices or mediums capable of operating as described herein. In some aspects, Storage 27 may include any features, functionalities, and embodiments of Memory 12, and vice versa, as applicable.

Processor 11 can communicate directly with cache memory 14 via a connection means such as a secondary bus which may also sometimes be referred to as a backside bus. In some embodiments, processor 11 can communicate with cache memory 14 using the system bus 5. Cache memory 14 may typically have a faster response time than main memory 12 and can include a type of memory which is considered faster than main memory 12 such as, for example, SRAM, BSRAM, or EDRAM. Cache memory includes any structure such as multilevel caches, for example. In some embodiments, processor 11 can communicate with one or more I/O devices 13 via a system bus 5. Various busses can be used to connect processor 11 to any of the I/O devices 13 such as a VESA VL bus, an ISA bus, an EISA bus, a MicroChannel Architecture (MCA) bus, a PCI bus, a PCI-X bus, a PCI-Express bus, a NuBus, and/or others. In some embodiments, processor 11 can communicate directly with I/O device 13 via HyperTransport, Rapid I/O, or InfiniBand. In further embodiments, local busses and direct communication can be mixed. For example, processor 11 can communicate with an I/O device 13 using a local interconnect bus and communicate with another I/O device 13 directly. Similar configurations can be used for any other components described herein.

Computing Device 70 may further include alternative memory such as a SD memory slot, a USB memory stick, an optical drive such as a CD-ROM drive, a CD-R/RW drive, a DVD-ROM drive or a BlueRay disc, a hard-drive, and/or any other device comprising non-volatile memory suitable for storing data or installing application programs. Computing Device 70 may further include a storage device 27 comprising any type or form of non-volatile memory for storing an operating system (OS) such as any type or form of Windows OS, Mac OS, Unix OS, Linux OS, Android OS, iPhone OS, mobile version of Windows OS, an embedded OS, or any other OS that can operate on Computing Device 70. Computing Device 70 may also include application programs 18, and/or data space 19 for

storing additional data or information. In some embodiments, alternative memory 16 can be used as or similar to storage device 27. Additionally, OS 17 and/or application programs 18 can be operable from a bootable medium such as, for example, a flash drive, a micro SD card, a bootable CD or DVD, and/or other bootable medium.

Application Program 18 (also referred to as program, computer program, application, script, code, or other suitable name or reference) comprises instructions that can provide functionality when executed by processor 11. Application program 18 can be implemented in a high-level procedural or object-oriented programming language, or in a low-level machine or assembly language. Any language used can be compiled, interpreted, or translated into machine language. Application program 18 can be deployed in any form including as a stand-alone program or as a module, component, subroutine, or other unit suitable for use in a computing system. Application program 18 does not necessarily correspond to a file in a file system. A program can be stored in a portion of a file that may hold other programs or data, in a single file dedicated to the program, or in multiple files (i.e. files that store one or more modules, sub programs, or portions of code, etc.). Application Program 18 can be delivered in various forms such as, for example, executable file, library, script, plugin, addon, applet, interface, console application, web application, application service provider (ASP)-type application, operating system, and/or other forms. Application program 18 can be deployed to be executed on one computing device or on multiple computing devices (i.e. cloud, distributed, or parallel computing, etc.), or at one site or distributed across multiple sites interconnected by a network or an interface. Examples of Application Program 18 include a computer game, a virtual world application, a graphics application, a media application, a word processing application, a spreadsheet application, a database application, a web browser, a forms-based application, a global positioning system (GPS) application, a 2D application, a 3D application, an operating system, a factory automation application, a device control application, a vehicle control application, and/or other application or program.

Network interface 25 can be utilized for interfacing Computing Device 70 with other devices via a network through a variety of connections including telephone lines, wired or wireless connections, LAN or WAN links (i.e. 802.11, T1, T3, 56 kb, X.25, etc.), broadband connections (i.e. ISDN, Frame Relay, ATM, etc.), or a combination thereof. Examples of networks include the Internet, an intranet, an extranet, a local area network (LAN), a wide area network (WAN), a personal area network (PAN), a home area network (HAN), a campus area network (CAN), a metropolitan area network (MAN), a global area network (GAN), a storage area network (SAN), virtual network, a virtual private network (VPN), a Bluetooth network, a wireless network, a wireless LAN, a radio network, a HomePNA, a power line communication network, a G.hn network, an optical fiber network, an Ethernet network, an active networking network, a client-server network, a peer-to-peer network, a bus network, a star network, a ring network, a mesh network, a star-bus network, a tree network, a hierarchical topology network, and/or other networks. Network interface 25 may include a built-in network adapter, network interface card, PCMCIA network card, card bus network adapter, wireless network adapter, Bluetooth network adapter, WiFi network adapter, USB network adapter, modem, and/or any other device suitable for interfacing Computing Device 70 with any type of network capable of communication and/or operations described herein.

I/O devices 13 may be present in various shapes or forms in Computing Device 70. Examples of I/O device 13 capable of input include a joystick, a keyboard, a mouse, a trackpad, a trackpoint, a trackball, a microphone, a drawing tablet, a glove, a tactile input device, a still or video camera, and/or other input device. Examples of I/O device 13 capable of output include a video display, a projector, a glasses, a speaker, a tactile output device, and/or

other output device. Examples of I/O device 13 capable of input and output include a touchscreen, a disk drive, an optical storage device, a modem, a network card, and/or other input/output device. I/O device 13 can be interfaced with processor 11 via an I/O port 15, for example. In some aspects, I/O device 13 can be a bridge between system bus 5 and an external communication bus such as a USB bus, an Apple Desktop Bus, an RS-232 serial connection, a SCSI bus, a FireWire bus, a FireWire 800 bus, an Ethernet bus, an AppleTalk bus, a Gigabit Ethernet bus, an Asynchronous Transfer Mode bus, a HIPPI bus, a Super HIPPI bus, a SerialPlus bus, a SCI/LAMP bus, a FibreChannel bus, a Serial Attached small computer system interface bus, and/or other bus.

An output interface (not shown) such as a graphical output interface, an acoustic output interface, a tactile output interface, a renderer, any device driver (i.e. audio, video, or other driver), and/or other output interface or system can be utilized to process output from elements of Computing Device 70 for conveyance on an output device such as Display 21. In some aspects, Display 21 or other output device itself may include an output interface for processing output from elements of Computing Device 70. Further, an input interface (not shown) such as a keyboard listener, a touchscreen listener, a mouse listener, any device driver (i.e. audio, video, keyboard, mouse, touchscreen, or other driver), and/or other input interface or system can be utilized to process input from Human-machine Interface 23 or other input device for use by elements of Computing Device 70. In some aspects, Human-machine Interface 23 or other input device itself may include an input interface for processing input for use by elements of Computing Device 70.

Computing Device 70 may include or be connected to multiple display devices 21. Display devices 21 can each be of the same or different type or form. Computing Device 70 and/or its elements comprise any type or form of suitable hardware, programs, or a combination thereof to support, enable, or provide for the connection and use of multiple display devices 21. In one example, Computing Device 70 includes any type or form of video adapter, video card, driver, and/or library to interface, communicate, connect, or otherwise use display devices 21. In some aspects, a video adapter may include multiple connectors to interface to multiple display devices 21. In other aspects, Computing Device 70 includes multiple video adapters, with each video adapter connected to one or more display devices 21. In some embodiments, Computing Device's 70 operating system can be configured for using multiple displays 21. In other embodiments, one or more display devices 21 can be provided by one or more other computing devices such as remote computing devices connected to Computing Device 70 via a network or an interface.

Computing Device 70 can operate under the control of operating system 17, which may support Computing Device's 70 basic functions, interface with and manage hardware resources, interface with and manage peripherals, provide common services for application programs, schedule tasks, and/or perform other functionalities. A modern operating system enables features and functionalities such as a high resolution display, graphical user interface (GUI), touchscreen, cellular network connectivity (i.e. mobile operating system, etc.), Bluetooth connectivity, WiFi connectivity, global positioning system (GPS) capabilities, mobile navigation, microphone, speaker, still picture camera, video camera, voice recorder, speech recognition, music player, video player, near field communication, personal digital assistant (PDA), and/or other features, functionalities, or applications. For example, Computing Device 70 can use any conventional operating system, any embedded operating system, any real-time operating system, any open source operating system, any video gaming operating system, any proprietary operating system, any online operating system, any operating system for mobile computing devices, or any other operating system

capable of running on Computing Device 70 and performing operations described herein. Example of operating systems include Windows XP, Windows 7, Windows 8, Windows 10, etc. manufactured by Microsoft Corporation of Redmond, Wash.; Mac OS, iPhone OS, etc. manufactured by Apple Computer of Cupertino, Calif.; OS/2 manufactured by International Business Machines of Armonk, N.Y.; Linux, a freely-available operating system distributed by Caldera Corp. of Salt Lake City, Utah; or any type or form of a Unix operating system, and/or others. Any operating systems such as the ones for Android devices can similarly be utilized.

Computing Device 70 can be implemented as or be part of various model architectures such as web services, distributed computing, grid computing, cloud computing, and/or other architectures. For example, in addition to the traditional desktop, server, or mobile operating system architectures, a cloud-based operating system can be utilized to provide the structure on which embodiments of the disclosure can be implemented. Other aspects of Computing Device 70 can also be implemented in the cloud without departing from the spirit and scope of the disclosure. For example, memory, storage, processing, and/or other elements can be hosted in the cloud. In some embodiments, Computing Device 70 can be implemented on multiple devices. For example, a portion of Computing Device 70 can be implemented on a mobile device and another portion can be implemented on wearable electronics.

Computing Device 70 can be or include any mobile device, a mobile phone, a smartphone (i.e. iPhone, Windows phone, Blackberry phone, Android phone, etc.), a tablet, a personal digital assistant (PDA), wearable electronics, implantable electronics, and/or other mobile device capable of implementing the functionalities described herein. Computing Device 70 can also be or include an embedded device, which can be any device or system with a dedicated function within another device or system. Embedded systems range from the simplest ones dedicated to one task with no user interface to complex ones with advanced user interface that may resemble modern desktop computer systems. Examples of devices comprising an embedded device include a mobile telephone, a personal digital assistant (PDA), a gaming device, a media player, a digital still or video camera, a pager, a television device, a set-top box, a personal navigation device, a global positioning system (GPS) receiver, a portable storage device (i.e. a USB flash drive, etc.), a digital watch, a DVD player, a printer, a microwave oven, a washing machine, a dishwasher, a gateway, a router, a hub, an automobile entertainment system, an automobile navigation system, a refrigerator, a washing machine, a factory automation device, an assembly line device, a factory floor monitoring device, a thermostat, an automobile, a factory controller, a telephone, a network bridge, and/or other devices. An embedded device can operate under the control of an operating system for embedded devices such as MicroC/OS-II, QNX, VxWorks, eCos, TinyOS, Windows Embedded, Embedded Linux, and/or other embedded device operating systems.

Various implementations of the disclosed devices, systems, and methods can be realized in digital electronic circuitry, integrated circuitry, logic gates, application specific integrated circuits (ASICs), field programmable gate arrays (FPGAs), computer hardware, firmware, programs, virtual machines, and/or combinations thereof including their structural, logical, and/or physical equivalents.

The disclosed devices, systems, and methods may include clients and servers. A client and server are generally, but not always, remote from each other and typically, but not always, interact via a network or an interface. The relationship of a client and server may arise by virtue of computer programs running on their respective computers and having a client-server relationship to each other, for example.

The disclosed devices, systems, and methods can be implemented in a computing system that includes a back end component, a middleware component, a front end component, or any combination thereof. The components of the system can be interconnected by any form or medium of digital data communication such as, for example, a network.

Computing Device 70 may include or be interfaced with a computer program product comprising instructions or logic encoded on a computer-readable medium. Such instructions or logic, when executed, may configure or cause one or more processors 11 to perform the operations and/or functionalities disclosed herein. For example, a computer program can be provided or encoded on a computer-readable medium such as an optical medium (i.e. DVD-ROM, etc.), flash drive, hard drive, any memory, firmware, or other medium. Machine-readable medium, computer-readable medium, or other such terms may refer to any computer program product, apparatus, and/or device for providing instructions and/or data to one or more programmable processors. As such, machine-readable medium includes any medium that can send and/or receive machine instructions as a machine-readable signal. Examples of a machine-readable medium include a volatile and/or non-volatile medium, a removable and/or non-removable medium, a communication medium, a storage medium, and/or other medium. A non-transitory machine-readable medium comprises all machine-readable media except for a transitory, propagating signal.

In some embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using an avatar's circumstances for autonomous avatar operation, or elements thereof, can be implemented entirely or in part in a device (i.e. microchip, circuitry, logic gates, electronic device, computing device, special or general purpose processor, etc.) or system that comprises (i.e. hard coded, internally stored, etc.) or is provided with (i.e. externally stored, etc.) instructions for implementing ACAAO functionalities. As such, the disclosed artificially intelligent devices, systems, and methods for learning and/or using an avatar's circumstances for autonomous avatar operation, or elements thereof, may include the processing, memory, storage, and/or other features, functionalities, and embodiments of Computing Device 70 or elements thereof. Such device or system can operate on its own (i.e. standalone device or system, etc.), be embedded in another device or system (i.e. an industrial machine, a robot, a vehicle, a toy, a smartphone, a television device, an appliance, and/or any other device or system capable of housing the elements needed for ACAAO functionalities), work in combination with other devices or systems, or be available in any other configuration. In other embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using an avatar's circumstances for autonomous avatar operation, or elements thereof, may include or be interfaced with Alternative Memory 16 that provides instructions for implementing ACAAO functionalities to one or more Processors 11. In further embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using an avatar's circumstances for autonomous avatar operation, or elements thereof, can be implemented entirely or in part as a computer program and executed by one or more Processors 11. Such program can be implemented in one or more modules or units of a single or multiple computer programs. Such program may be able to attach to or interface with, inspect, and/or take control of another application program to implement ACAAO functionalities. In further embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using an avatar's circumstances for autonomous avatar operation, or elements thereof, can be implemented as a network, web, distributed, cloud, or other such application accessed on one or more remote computing devices (i.e. servers, cloud, etc.) via Network Interface 25, such remote computing devices including processing capabilities and instructions for implementing

ACAAO functionalities. In further embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using an avatar's circumstances for autonomous avatar operation, or elements thereof, can be (1) attached to or interfaced with any computing device or application program, (2) included as a feature of an operating system, (3) built (i.e. hard coded, etc.) into any computing device or application program, and/or (4) available in any other configuration to provide its functionalities.

In some embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using an avatar's circumstances for autonomous avatar operation, or elements thereof, can be implemented at least in part in a computer program such as Java application or program. Java provides a robust and flexible environment for application programs including flexible user interfaces, robust security, built-in network protocols, powerful application programming interfaces, database or DBMS connectivity and interfacing functionalities, file manipulation capabilities, support for networked applications, and/or other features or functionalities. Application programs based on Java can be portable across many devices, yet leverage each device's native capabilities. Java supports the feature sets of most smartphones and a broad range of connected devices while still fitting within their resource constraints. Various Java platforms include virtual machine features comprising a runtime environment for application programs. Java platforms provide a wide range of user-level functionalities that can be implemented in application programs such as displaying text and graphics, playing and recording audio content, displaying and recording visual content, communicating with another computing device, and/or other functionalities. It should be understood that the disclosed artificially intelligent devices, systems, and methods for learning and/or using an avatar's circumstances for autonomous avatar operation, or elements thereof, are programming language, platform, and operating system independent. Examples of programming languages that can be used instead of or in addition to Java include C, C++, Cobol, Python, Java Script, Tcl, Visual Basic, Pascal, VB Script, Perl, PHP, Ruby, and/or other programming languages capable of implementing the functionalities described herein.

Where a reference to a specific file or file type is used herein, other files or file types can be substituted.

Where a reference to a data structure is used herein, it should be understood that any variety of data structures can be used such as, for example, array, list, linked list, doubly linked list, queue, tree, heap, graph, map, grid, matrix, multi-dimensional matrix, table, database, database management system (DBMS), file, neural network, and/or any other type or form of a data structure including a custom one. A data structure may include one or more fields or data fields that are part of or associated with the data structure. A field or data field may include a data, an object, a data structure, and/or any other element or a reference/pointer thereto. A data structure can be stored in one or more memories, files, or other repositories. A data structure and/or elements thereof, when stored in a memory, file, or other repository, may be stored in a different arrangement than the arrangement of the data structure and/or elements thereof. For example, a sequence of elements can be stored in an arrangement other than a sequence in a memory, file, or other repository.

Where a reference to a repository is used herein, it should be understood that a repository may be or include one or more files or file systems, one or more storage locations or structures, one or more storage systems, one or more memory locations or structures, and/or other file, storage, memory, or data arrangements.

Where a reference to an interface is used herein, it should be understood that the interface comprises any hardware, device, system, program, method, and/or combination thereof that enable direct or operative coupling, connection, and/or interaction of the elements between which the interface is indicated. A line or arrow shown in the

figures between any of the depicted elements comprises such interface. Examples of an interface include a direct connection, an operative connection, a wired connection (i.e. wire, cable, etc.), a wireless connection, a device, a network, a bus, a circuit, a firmware, a driver, a bridge, a program, a combination thereof, and/or others.

Where a reference to an element coupled or connected to another element is used herein, it should be understood that the element may be in communication or other interactive relationship with the other element. Furthermore, an element coupled or connected to another element can be coupled or connected to any other element in alternate implementations. Terms coupled, connected, interfaced, or other such terms may be used interchangeably herein depending on context.

Where a reference to an element matching another element is used herein, it should be understood that the element may be equivalent or similar to the other element. Therefore, the term match or matching can refer to total equivalence or similarity depending on context.

Where a reference to a device is used herein, it should be understood that the device may include or be referred to as a system, and vice versa depending on context, since a device may include a system of elements and a system may be embodied in a device.

Where a reference to a collection of elements is used herein, it should be understood that the collection of elements may include one or more elements. In some aspects or contexts, a reference to a collection of elements does not imply that the collection is an element itself.

Where a mention of a function, method, routine, subroutine, or other such procedure is used herein, it should be understood that the function, method, routine, subroutine, or other such procedure comprises a call, reference, or pointer to the function, method, routine, subroutine, or other such procedure.

Where a mention of data, object, data structure, item, element, or thing is used herein, it should be understood that the data, object, data structure, item, element, or thing comprises a reference or pointer to the data, object, data structure, item, element, or thing.

Referring to Fig. 2, an embodiment of Computing Device 70 comprising Unit for Learning and/or Using an Avatar's Circumstances for Autonomous Avatar Operation (ACAAO Unit 100) is illustrated. Computing Device 70 also comprises interconnected Processor 11, Display 21, Human-machine Interface 23, Memory 12, and Storage 27. Processor 11 includes or executes Application Program 18 comprising Avatar 605 and/or one or more Objects 615. ACAAO Unit 100 comprises interconnected Artificial Intelligence Unit 110, Acquisition Interface 120, Modification Interface 130, and Object Processing Unit 140. Other additional elements can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate embodiments.

In one example, the teaching presented by the disclosure can be implemented in a device or system for learning and/or using an avatar's circumstances for autonomous avatar operating. The device or system may include a processor circuit (i.e. Processor 11, etc.) configured to execute instruction sets (i.e. Instruction Sets 526, etc.) of an application. The device or system may further include a memory unit (i.e. Memory 12, etc.) configured to store data. The device or system may further include an artificial intelligence unit (i.e. Artificial Intelligence Unit 110, etc.). The artificial intelligence unit may be configured to receive a first collection of object representations (i.e. Collection of Object Representations 525, etc.), the first collection of object representations including one or more object representations representing one or more objects of the application. The artificial intelligence unit may also be configured to receive a first one or more instruction sets for operating an avatar of the application. The artificial

intelligence unit may also be configured to learn the first collection of object representations correlated with the first one or more instruction sets for operating the avatar of the application. The artificial intelligence unit may also be configured to receive a new collection of object representations, the new collection of object representations including one or more object representations representing one or more objects of the application. The artificial intelligence unit may also be configured to anticipate the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may also be configured to cause the processor circuit to execute the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the avatar of the application performs one or more operations defined by the first one or more instruction sets for operating the avatar of the application correlated with the first collection of object representations. Any of the operations of the aforementioned elements can be performed repeatedly and/or in different orders in alternate embodiments. In some embodiments, a collection of object representations may include or be substituted with a stream of collections of object representations. In some embodiments of applications that do not comprise an avatar or rely on avatar for their operation, the teaching presented by the disclosure can be implemented in a device or system for learning and/or using an application's circumstances for autonomous application operating. In such embodiments, an instruction set for operating an avatar of an application may include or be substituted with an instruction set for operating an application. Other additional elements can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate embodiments. The disclosed devices and systems may include any actions or operations of any of the disclosed methods such as methods 9100, 9200, 9300, 9400, 9500, 9600, and/or others (all later described).

User 50 (also referred to simply as user or other suitable name or reference) comprises a human user or non-human user. A non-human User 50 includes any device, system, program, and/or other mechanism for operating or controlling Application Program 18, Avatar 605, Computing Device 70, and/or elements thereof. For example, User 50 may issue an operating direction to Application Program 18 responsive to which Application Program's 18 instructions or instruction sets may be executed by Processor 11 to perform a desired operation with/on Avatar 605. User's 50 operating directions comprise any user inputted data (i.e. values, text, symbols, etc.), directions (i.e. move right, move up, move forward, copy an item, click on a link, etc.), instructions or instruction sets (i.e. manually inputted instructions or instruction sets, etc.), and/or other inputs or information. A non-human User 50 can utilize more suitable interfaces instead of, or in addition to, Human-machine Interface 23 and/or Display 21 for controlling Application Program 18, Avatar 605, Computing Device 70, and/or elements thereof. Examples of such interfaces include an application programming interface (API), bridge (i.e. bridge between applications, devices, or systems, etc.), driver, socket, direct or operative connection, handle, function/routine/subroutine, and/or other interfaces.

Avatar 605 may be or comprises an object of Application Program 18. While Avatar 605 may include any features, functionalities, and embodiments of Object 615 (later described), Avatar 605 is distinguished herein to portray the relationships and/or interactions between Avatar 605 and other Objects 615 of Application Program 18. In some aspects, Avatar 605 includes a User 50-controllable object of Application Program 18. Avatar 605 may,

therefore, be a representation of User 50 or of User's 50 actions, thoughts, and/or other expressions. In some designs, Avatar 605 includes a 2D model, a 3D model, a 2D shape (i.e. point, line, square, rectangle, circle, triangle, etc.), a 3D shape (i.e. cube, sphere, etc.), a graphical user interface (GUI) element, a picture, and/or other models, shapes, elements, or objects. Avatar 605 may perform one or more operations within Application Program 18. For example, Avatar 605 may perform operations including moving, maneuvering, jumping, running, shooting, and/or other operations within a game or virtual world Application Program 18. While all possible variations of operations on/by/with Avatar 605 are too voluminous to list and limited only by Avatar's 605 and/or Application Program's 18 design, and/or User's 50 utilization, other operations on/by/with Avatar 605 are within the scope of this disclosure.

Object Processing Unit 140 comprises the functionality for obtaining information of interest on objects of Application Program 18, and/or other functionalities. As such, Object Processing Unit 140 can be used to obtain objects and/or their properties in Avatar's 605 surrounding within Application Program 18. Avatar's 605 surrounding may include or be defined by Area of Interest 450 (later described), part of Application Program 18 that is shown to User 50 (i.e. on a display, via a graphical user interface, etc.), the entire Application Program 18, any part of Application Program 18, and/or other techniques. In some embodiments, Object Processing Unit 140 comprises the functionality for creating or generating Collection of Object Representations 525 (also referred to as Coll of Obj Rep or other suitable name or reference) and storing one or more Object Representations 625 (also referred to simply as object representations, representations of objects, or other suitable name or reference), Object Properties 630 (also referred to simply as object properties or other suitable name or reference), and/or other elements or information into the Collection of Object Representations 525. As such, Collection of Object Representations 525 comprises the functionality for storing one or more Object Representations 625, Object Properties 630, and/or other elements or information. In some designs, Object Representation 625 may include a representation of an object (i.e. Object 615 [later described], etc.) in Avatar's 605 surrounding within Application Program 18. As such, Object Representation 625 may include any information related to an object. In other designs, Object Representation 625 may include or be replaced with an object itself, in which case Object Representation 625 as an element can be omitted. In some aspects, Collection of Object Representations 525 includes one or more Object Representations 625, Object Properties 630, and/or other elements or information related to objects in Avatar's 605 surrounding at a particular time. Collection of Object Representations 525 may, therefore, include knowledge (i.e. unit of knowledge, etc.) of Avatar's 605 circumstance including objects with various properties at a particular time. In some designs, a Collection of Object Representations 525 may include or be associated with a time stamp (not shown), order (not shown), or other time related information. For example, one Collection of Object Representations 525 may be associated with time stamp t1, another Collection of Object Representations 525 may be associated with time stamp t2, and so on. Time stamps t1, t2, etc. may indicate the times of generating Collections of Object Representations 525, for instance. In other embodiments, Object Processing Unit 140 comprises the functionality for creating or generating a stream of Collections of Object Representations 525. A stream of Collections of Object Representations 525 may include one Collection of Object Representations 525 or a group, sequence, or other plurality of Collections of Object Representations 525. In some aspects, a stream of Collections of Object Representations 525 includes one or more Collections of Object Representations 525, and/or other elements or information related to objects in Avatar's 605 surrounding over time. A stream of Collections of Object Representations 525 may, therefore, include knowledge (i.e. unit of knowledge, etc.) of Avatar's 605 circumstance

including objects with various properties over time. As circumstances including objects with various properties in Avatar's 605 surrounding change (i.e. objects and/or their properties change, move, act, transform, etc.) over time, this change may be captured in a stream of Collections of Object Representations 525. In some designs, each Collection of Object Representations 525 in a stream may include or be associated with the aforementioned time stamp, order, or other time related information. For example, one Collection of Object Representations 525 in a stream may be associated with order 1, a next Collection of Object Representations 525 in the stream may be associated with order 2, and so on. Orders 1, 2, etc. may indicate the orders or places of Collections of Object Representations 525 within a stream (i.e. sequence, etc.), for instance. In some implementations, Object Processing Unit 140 and/or any of its elements or functionalities can be included or embedded in Computing Device 70, Processor 11, Application Program 18, and/or other elements. Object Processing Unit 140 can be provided in any suitable configuration.

Examples of Objects 615 (also referred to simply as objects, etc.) include models of a person, animal, tree, rock, building, vehicle, and/or others in a context of a computer game, virtual world, 3D or 2D graphics Application Program 18, and/or others. More generally, examples of Objects 615 include a 2D model, a 3D model, a 2D shape (i.e. point, line, square, rectangle, circle, triangle, etc.), a 3D shape (i.e. cube, sphere, etc.), a graphical user interface (GUI) element, a form element (i.e. text field, radio button, push button, check box, etc.), a data or database element, a spreadsheet element, a link, a picture, a text (i.e. character, word, etc.), a number, and/or others in a context of a web browser, a media application, a word processing application, a spreadsheet application, a database application, a forms-based application, an operating system, a device/system control application, and/or others. Object 615 may perform operations within Application Program 18. In one example, a person Object 615 may perform operations including moving, maneuvering, jumping, running, shooting, and/or other operations within a computer game, virtual world, and/or 3D or 2D graphics Application Program 18. In another example, a character Object 615 may perform operations including appearing (i.e. when typed, etc.), disappearing (i.e. when deleted, etc.), formatting (i.e. bolding, italicizing, underlining, coloring, resizing, etc.), and/or other operations within a word processing Application Program 18. In a further example, a picture Object 615 may perform operations including resizing, repositioning, rotating, deforming, and/or other operations within a graphics Application Program 18. While all possible variations of operations on/by/with Object 615 are too voluminous to list and limited only by Object's 615 and/or Application Program's 18 design, and/or User's 50 utilization, other operations on/by/with Object 615 are within the scope of this disclosure. In some aspects, any part of Object 615 can be identified as an Object 615 itself. For instance, instead of or in addition to identifying a building as an Object 615, a window, door, roof, and/or other parts of the building can be identified as Objects 615. In general, Object 615 may include any object or part thereof that can be obtained or recognized.

Examples of Object Properties 630 (i.e. also referred to simply as object properties, etc.) include existence of Object 615, type of Object 615 (i.e. person, animal, tree, rock, building, vehicle, etc.), identity of Object 615 (i.e. name, identifier, etc.), distance of Object 615, bearing/angle of Object 615, location of Object 615 (i.e. distance and bearing/angle from a known point, coordinates, etc.), shape/size of Object 615 (i.e. scale, height, width, depth, computer model, etc.), activity of Object 615 (i.e. motion, gestures, etc.), and/or other properties of Object 615. Type of Object 615, for example, may include any classification of objects ranging from detailed such as person, animal, tree, rock, building, vehicle, etc. to generalized such as biological object, nature object, manmade object, etc., or

models thereof, including their sub-types. Location of Object 615, for example, can include a relative location such as one defined by distance and bearing/angle from a known point or location (i.e. Avatar 605 location, etc.). Location of Object 615, for example, can also include absolute location such as one defined by object coordinates. In general, Object Property 630 may include any attribute of Object 615 (i.e. existence of Object 615, type of Object 615, identity of Object 615, shape/size of Object 615, etc.), any relationship of Object 615 with Avatar 605, other Object 615, or the environment (i.e. distance of Object 615, bearing/angle of Object 615, friend/foe relationship, etc.), and/or other information related to Object 615.

In some embodiments, Object Processing Unit 140 can be utilized for obtaining properties of Objects 615 in Avatar's 605 surrounding within Application Program 18. In some designs, an engine, environment, or other system used to implement Application Program 18 includes functions for providing properties or other information on Objects 615. Object Processing Unit 140 can obtain Object Properties 630 by utilizing the functions. In some aspects, existence of Object 615 in a 2D or 3D engine or environment can be obtained by utilizing functions such as GameObject.FindObjectsOfType(GameObject), GameObject.FindGameObjectsWithTag("TagN"), or GameObject.Find("ObjectN") in Unity 3D Engine; GetAllActorsOfClass() or IsActorInitialized() in Unreal Engine; and/or other functions, procedures, or methods in other 2D or 3D engines or environments. In other aspects, type or other classification (i.e. person, animal, tree, rock, building, vehicle, etc.) of Object 615 in a 2D or 3D engine or environment can be obtained by utilizing functions such as GetClassName(ObjectN) or ObjectN.getType() in Unity 3D Engine; ActorN.GetClass() in Unreal Engine; ObjectN.getClassName() or ObjectN.getType() in Torque 3D Engine; and/or other functions, procedures, or methods in other 2D or 3D engines or environments. In further aspects, identity of Object 615 in a 2D or 3D engine or environment can be obtained by utilizing functions such as ObjectN.name or ObjectN.GetInstanceID() in Unity 3D Engine; ActorN.GetObjectName() or ActorN.GetUniqueID() in Unreal Engine; ObjectN.getName() or ObjectN.getID() in Torque 3D Engine; and/or other functions, procedures, or methods in other 2D or 3D engines or environments. In further aspects, distance of Object 615 relative to Avatar 605 in a 2D or 3D engine or environment can be obtained by utilizing functions such as VectorN.Distance(ObjectA.transform.position, ObjectB.transform.position) in Unity 3D Engine; GetDistanceTo(ActorA, ActorB) in Unreal Engine; VectorDist(VectorA, VectorB) or VectorDist(ObjectA.getPosition(),ObjectB.getPosition()) in Torque 3D Engine; and/or other functions, procedures, or methods in other 2D or 3D engines or environments. In further aspects, angle, bearing, or direction of Object 615 relative to Avatar 605 in a 2D or 3D engine or environment can be obtained by utilizing functions such as ObjectB.transform.position - ObjectA.transform.position in Unity 3D Engine; FindLookAtRotation(TargetVector, StartVector) or ActorB->GetActorLocation() - ActorA->GetActorLocation() in Unreal Engine; ObjectB->getPosition() - ObjectA->getPosition() in Torque 3D Engine; and/or other functions, procedures, or methods in other 2D or 3D engines or environments. In further aspects, location of Object 615 in a 2D or 3D engine or environment can be obtained by utilizing functions such as ObjectN.transform.position in Unity 3D Engine; ActorN.GetActorLocation() in Unreal Engine; ObjectN.getPosition() in Torque 3D Engine; and/or other similar functions, procedures, or methods in other 2D or 3D engines or environments. In another example, location (i.e. coordinates, etc.) of Object 615 on a screen can be obtained by utilizing WorldToScreen() or other similar function or method in various 2D or 3D engines or environments. In some designs, distance, angle/bearing, and/or other properties of Object 615 relative to Avatar 605 can then be calculated, inferred, derived, or estimated from Object's 615 and Avatar's 605 location information.

Object Processing Unit 140 may include computational functionalities to perform such calculations, inferences, derivations, or estimations by utilizing, for example, geometry, trigonometry, Pythagorean theorem, and/or other theorems, formulas, or disciplines. In further aspects, shape/size of Object 615 in a 2D or 3D engine or environment can be obtained by utilizing functions such as Bounds.size, ObjectN.transform.localScale, or ObjectN.transform.lossyScale in Unity 3D Engine; ActorN.GetActorBounds(), ActorN.GetActorScale(), or ActorN.GetActorScale3D() in Unreal Engine; ObjectN.getObjectBox() or ObjectN.getScale() in Torque 3D Engine; and/or other similar functions, procedures, or methods in other 2D or 3D engines or environments. In some designs, detailed shape of Object 615 can be obtained by accessing the object's mesh or computer model. In general, any of the aforementioned and/or other properties of Object 615 can be obtained by accessing a scene graph or other data structure used for organizing objects in a particular engine or environment, finding a specific Object 615, and obtaining or reading any property from the Object 615. Such accessing can be performed by using the engine's or environment's functions for accessing objects in the scene graph or other data structure or by directly accessing the scene graph or other data structure. In some designs, functions and/or other instructions for obtaining properties or other information on Objects 615 of Application Program 18 can be inserted or utilized in Application Program's 18 source code. In other designs, functions and/or other instructions for obtaining properties or other information on Objects 615 of Application Program 18 can be inserted into Application Program 18 through manual, automatic, dynamic, or just-in-time (JIT) instrumentation (later described). In further designs, functions and/or other instructions for providing properties or other information on Objects 615 of Application Program 18 can be inserted into Application Program 18 through utilizing dynamic code, dynamic class loading, reflection, and/or other functionalities of a programming language or platform; utilizing dynamic, interpreted, and/or scripting programming languages; utilizing metaprogramming; and/or utilizing other techniques (later described). Object Processing Unit 140 may include any features, functionalities, and embodiments of Acquisition Interface 120, Modification Interface 130, and/or other elements. One of ordinary skill in art will understand that the aforementioned techniques for obtaining objects and/or their properties are described merely as examples of a variety of possible implementations, and that while all possible techniques for obtaining objects and/or their properties are too voluminous to describe, other techniques for obtaining objects and/or their properties known in art are within the scope of this disclosure. It should be noted that Unity 3D Engine, Unreal Engine, and Torque 3D Engine are used merely as examples of a variety of engines, environments, or systems that can be used to implement Application Program 18 and any of the aforementioned functionalities may be provided in other engines, environments, or systems. Also, in some embodiments, Application Program 18 may not use any engine, environment, or system for its implementation, in which case the aforementioned functionalities can be implemented within Application Program 18. In general, the disclosed devices, systems, and methods are independent of the engine, environment, or system used to implement Application Program 18.

In some embodiments of Application Programs 18 that do not comprise Avatar 605 or rely on Avatar 605 for their operation, Object Processing Unit 140 may obtain objects and/or their properties in Application Program 18 or a part thereof. For example, Object Processing Unit 140 can obtain objects and/or their properties in the entire Application Program 18, a part of Application Program 18 that is shown to User 50 (i.e. on a display, via a graphical user interface, etc.), or any part or area of interest (later described) of Application Program 18. In such embodiments, Object Processing Unit 140 can create or generate Collections of Object Representations 525 or

streams of Collections of Object Representations 525 comprising knowledge (i.e. unit of knowledge, etc.) of Application Program's 18 circumstances including objects with various properties. It should be noted that a reference to Avatar 605 may include or be substituted with a reference to Application Program 18 and/or other processing element, and vice versa, depending on context (i.e. whether Avatar's 605 or Application Program's 18 operation is being learned and/or used, etc.). Also, a reference to operating and/or autonomous operating of Avatar 605 may include or be substituted with a reference to operating and/or autonomous operating of Application Program 18 and/or other processing element depending on context.

Referring to Fig. 3, an embodiment of utilizing Picture Renderer 91 and Picture Recognizer 92 is illustrated.

Picture Renderer 91 comprises the functionality for rendering or generating one or more digital pictures, and/or other functionalities. Picture Renderer 91 comprises the functionality for rendering or generating one or more digital pictures of Application Program 18. In some aspects, as a camera is used to capture pictures of a physical environment, Picture Renderer 91 can be used to render or generate pictures of a computer modeled or represented environment. As such, Picture Renderer 91 can be used to render or generate views of Application Program 18. In some designs, Picture Renderer 91 can be used to render or generate one or more digital pictures depicting a view of an Avatar's 605 visual surrounding in a 3D Application Program 18 (i.e. 3D computer game, virtual world application, CAD application, etc.). In one example, a view may include a first-person view or perspective such as a view through an avatar's eyes that shows objects around the avatar, but does not typically show the avatar itself. First-person view may sometimes include the avatar's hands, feet, other body parts, and/or objects that the avatar is holding. In another example, a view may include a third-person view or perspective such as a view that shows an avatar as well as objects around the avatar from an observer's point of view. In a further example, a view may include a view from a front of an avatar. In a further example, a view may include a view from a side of an avatar. In a further example, a view may include any stationary or movable view such as a view through a simulated camera in a 3D Application Program 18. In other designs, Picture Renderer 91 can be used to render or generate one or more digital pictures depicting a view of a 2D Application Program 18. In one example, a view may include a screenshot or portion thereof of a 2D Application Program 18. In a further example, a view may include an area of interest of a 2D Application Program 18. In a further example, a view may include a top-down view of a 2D Application Program 18. In a further example, a view may include a side-on view of a 2D Application Program 18. Any other view can be utilized in alternate designs. Any view utilized in a 3D Application Program 18 can similarly be utilized in a 2D Application Program 18 as applicable, and vice versa. In some implementations, Picture Renderer 91 may include any graphics processing device, apparatus, system, or application that can render or generate one or more digital pictures from a computer (i.e. 3D, 2D, etc.) model or representation. In some aspects, rendering, when used casually, may refer to rendering or generating one or more digital pictures from a computer model or representation, providing the one or more digital pictures to a display device, and/or displaying of the one or more digital pictures on a display device. In some embodiments, Picture Renderer 91 can be a program executing or operating on Processor 11. In one example, Picture Renderer 91 can be provided in a rendering engine such as Direct3D, OpenGL, Mantle, and/or other programs or systems for rendering or processing 3D or 2D graphics. In other embodiments, Picture Renderer 91 can be part of, embedded into, or built into Processor 11. In further embodiments, Picture Renderer 91 can be a hardware element coupled to Processor 11 and/or other elements. In further embodiments, Picture Renderer 91 can be a program or hardware element that is part of or embedded into another element. In one

example, a graphics card and/or its graphics processing unit (i.e. GPU, etc.) may typically include Picture Renderer 91. In another example, ACAAO Unit 100 may include Picture Renderer 91. In a further example, Application Program 18 may include Picture Renderer 91. In general, Picture Renderer 91 can be implemented in any suitable configuration to provide its functionalities. Picture Renderer 91 may render or generate one or more digital pictures or streams of digital pictures (i.e. motion pictures, video, etc.) in various formats examples of which include JPEG, GIF, TIFF, PNG, PDF, MPEG, AVI, FLV, MOV, RM, SWF, WMV, DivX, and/or others. In some designs, Picture Renderer 91 can render or generate different digital pictures of Avatar's 605 visual surrounding or of views of Application Program 18 for displaying on Display 21 and for facilitating object recognition functionalities herein. For example, a third-person view may be displayed on Display 21 for User 50 to see and a first-person view may be used to facilitate object recognition functionalities herein. In some implementations of non-graphical Application Programs 18 such as simulations, calculations, and/or others, Picture Renderer 91 may render or generate one or more digital pictures of Avatar's 605 visual surrounding or of views of Application Program 18 to facilitate object recognition functionalities herein where the one or more digital pictures are never displayed. In some aspects, instead of or in addition to Picture Renderer 91, one or more digital pictures of Avatar's 605 visual surrounding or of views of Application Program 18 can be obtained from any element of a computing device or system that can provide such digital pictures. Examples of such elements include a graphics circuit, a graphics system, a graphics driver, a graphics interface, and/or others.

Picture Recognizer 92 comprises the functionality for detecting or recognizing objects and/or their properties in visual data, and/or other disclosed functionalities. Visual data includes digital motion pictures, digital still pictures, and/or other visual data. Examples of file formats that can be utilized to store visual data include JPEG, GIF, TIFF, PNG, PDF, MPEG, AVI, FLV, MOV, RM, SWF, WMV, DivX, and/or other file formats. In some designs, Picture Recognizer 92 can be used for detecting or recognizing objects and/or their properties in one or more digital pictures from Picture Renderer 91. For example, Picture Recognizer 92 can be utilized in detecting or recognizing existence of an object, type of an object, identity of an object, shape/size of an object, activity of an object, and/or other properties of an object. In general, Picture Recognizer 92 can be used for any operation supported by Picture Recognizer 92. Picture Recognizer 92 may detect or recognize an object and/or its properties as well as track the object and/or its properties in one or more digital pictures or streams of digital pictures (i.e. motion pictures, video, etc.). In the case of a person, Picture Recognizer 92 may detect or recognize a human head or face, upper body, full body, or portions/combinations thereof. In some aspects, Picture Recognizer 92 may detect or recognize objects and/or their properties from a digital picture by comparing regions of pixels from the digital picture with collections of pixels comprising known objects and/or their properties. The collections of pixels comprising known objects and/or their properties can be learned or manually, programmatically, or otherwise defined. The collections of pixels comprising known objects and/or their properties can be stored in any data structure or repository (i.e. one or more files, database, etc.) that resides locally on Computing Device 70, or remotely on a remote computing device (i.e. server, cloud, etc.) accessible over a network or an interface. In other aspects, Picture Recognizer 92 may detect or recognize objects and/or their properties from a digital picture by comparing features (i.e. lines, edges, ridges, corners, blobs, regions, etc.) of the digital picture with features of known objects and/or their properties. The features of known objects and/or their properties can be learned or manually, programmatically, or otherwise defined. The features of known objects and/or their properties can be stored in any data structure or repository (i.e. neural

network, one or more files, database, etc.) that resides locally on Computing Device 70, or remotely on a remote computing device (i.e. server, cloud, etc.) accessible over a network or an interface. Typical steps or elements in a feature oriented picture recognition include pre-processing, feature extraction, detection/segmentation, decision-making, and/or others, or a combination thereof, each of which may include its own sub-steps or sub-elements depending on the application. In further aspects, Picture Recognizer 92 may detect or recognize multiple objects and/or their properties from a digital picture using the aforementioned pixel or feature comparisons, and/or other detection or recognition techniques. For example, a picture may depict two objects in two of its regions both of which Picture Recognizer 92 can detect simultaneously. In further aspects, where objects and/or their properties span multiple pictures, Picture Recognizer 92 may detect or recognize objects and/or their properties by applying the aforementioned pixel or feature comparisons and/or other detection or recognition techniques over a stream of digital pictures (i.e. motion picture, video, etc.). For example, once an object is detected in a digital picture (i.e. frame, etc.) of a stream of digital pictures (i.e. motion picture, video, etc.), the region of pixels comprising the detected object or the object's features can be searched in other pictures of the stream of digital pictures, thereby tracking the object through the stream of digital pictures. In further aspects, Picture Recognizer 92 may detect or recognize an object's activities by identifying and/or analyzing differences between a detected region of pixels of one picture (i.e. frame, etc.) and detected regions of pixels of other pictures in a stream of digital pictures. For example, a region of pixels comprising a person's face can be detected in multiple consecutive pictures of a stream of digital pictures (i.e. motion picture, video, etc.). Differences among the detected regions of the consecutive pictures may be identified in the mouth part of the person's face to indicate smiling or speaking activity. In further aspects, Picture Recognizer 92 may detect or recognize objects and/or their properties using one or more artificial neural networks, which may include statistical techniques. Examples of artificial neural networks that can be used in Picture Recognizer 92 include convolutional neural networks (CNNs), time delay neural networks (TDNNs), deep neural networks, and/or others. In one example, picture recognition techniques and/or tools involving convolutional neural networks may include identifying and/or analyzing tiled and/or overlapping regions or features of a digital picture, which may then be used to search for pictures with matching regions or features. In another example, features of different convolutional neural networks responsible for spatial and temporal streams can be fused to detect objects and/or their properties in streams of digital pictures (i.e. motion pictures, videos, etc.). In general, Picture Recognizer 92 may include any machine learning, deep learning, and/or other artificial intelligence techniques. Any other techniques known in art can be utilized in Picture Recognizer 92. For example, thresholds for similarity, statistical techniques, and/or optimization techniques can be utilized to determine a match in any of the above-described detection or recognition techniques.

Various aspects or properties of digital pictures or pixels can be taken into account by Picture Recognizer 92 in any of the recognizing or comparisons. Examples of such aspects or properties include color adjustment, size adjustment, content manipulation, transparency (i.e. alpha channel, etc.), use of mask, and/or others. In some implementations, as digital pictures can be captured or generated by various equipment, in various environments, and under various lighting conditions, Picture Recognizer 92 can adjust lighting or color of pixels or otherwise manipulate pixels before or during comparison. Lighting or color adjustment (also referred to as gray balance, neutral balance, white balance, etc.) may generally include manipulating or rebalancing the intensities of the colors (i.e. red, green, and/or blue if RGB color model is used, etc.) of one or more pixels. For example, Picture Recognizer

92 can adjust lighting or color of some or all pixels of one picture to make it more comparable to another picture. Picture Recognizer 92 can also incrementally adjust the pixels such as increasing or decreasing the red, green, and/or blue pixel values by a certain amount in each cycle of comparisons in order to find a similarity or match at one of the incremental adjustment levels. Any of the publically available, custom, or other lighting or color adjustment techniques or programs can be utilized such as color filters, color balancing, color correction, and/or others. In other implementations, Picture Recognizer 92 can resize or otherwise transform a digital picture before or during comparison. Such resizing or transformation may include increasing or decreasing the number of pixels of a digital picture. For example, Picture Recognizer 92 can increase or decrease the size of a digital picture proportionally (i.e. increase or decrease length and/or width keeping aspect ratio constant, etc.) to equate its size with the size of another digital picture. Picture Recognizer 92 can also incrementally resize a digital picture such as increasing or decreasing the size of the digital picture proportionally by a certain amount in each cycle of comparisons in order to find a similarity or match at one of the incremental sizes. Any of the publically available, custom, or other digital picture resizing techniques or programs can be utilized such as nearest-neighbor interpolation, bilinear interpolation, bicubic interpolation, and/or others. In further implementations, Picture Recognizer 92 can manipulate content (i.e. all pixels, one or more regions, one or more depicted objects, etc.) of a digital picture before or during comparison. Such content manipulation may include moving, centering, aligning, resizing, transforming, and/or otherwise manipulating content of a digital picture. For example, Picture Recognizer 92 can move, center, or align content of one picture to make it more comparable to another picture. Any of the publically available, custom, or other digital picture manipulation techniques or programs can be utilized such as pixel moving, warping, distorting, aforementioned interpolations, and/or others. In further implementations, in digital pictures comprising transparency features or functionalities, Picture Recognizer 92 can utilize a threshold for acceptable number or percentage transparency difference. Alternatively, transparency can be applied to one or more pixels of a digital picture and color difference may then be determined between compared pixels taking into account the transparency related color effect. Alternatively, transparent pixels can be excluded from comparison. In further implementations, certain regions or subsets of pixels can be ignored or excluded during comparison using a mask. In general, any region or subset of a picture determined to contain no content of interest can be excluded from comparison using a mask. Examples of such regions or subsets include background, transparent or partially transparent regions, regions comprising insignificant content, or any arbitrary region or subset. Picture Recognizer 92 can perform any other pre-processing or manipulation of digital pictures or pixels before or during recognizing or comparison.

In some exemplary embodiments, object recognition techniques and/or tools such as OpenCV (Open Source Computer Vision) library, CamFind API, Kooaba, 6px API, Dextro API, and/or others can be utilized for detecting or recognizing objects and/or their properties in digital pictures. In some aspects, picture recognition techniques and/or tools involve identifying and/or analyzing features such as lines, edges, ridges, corners, blobs, regions, and/or their relative positions, sizes, shapes, etc., which may then be used to search for pictures with matching features. For example, OpenCV library can detect an object (i.e. person, animal, vehicle, rock, etc.) and/or its properties in one or more digital pictures from Picture Renderer 91 or stored in an electronic repository, which can then be utilized in ACAAO Unit 100, Artificial Intelligence Unit 110, and/or other elements. In other exemplary embodiments, facial recognition techniques and/or tools such as OpenCV (Open Source Computer Vision) library,

Animetrics FaceR API, Lambda Labs Facial Recognition API, Face++ SDK, Neven Vision (also known as N-Vision) Engine, and/or others can be utilized for detecting or recognizing faces in digital pictures. In some aspects, facial recognition techniques and/or tools involve identifying and/or analyzing facial features such as the relative position, size, and/or shape of the eyes, nose, cheekbones, jaw, etc., which may then be used to search for pictures with matching features. For example, FaceR API can detect a person's face in one or more digital pictures from Picture Renderer 91 or stored in an electronic repository, which can then be utilized in ACAAO Unit 100, Artificial Intelligence Unit 110, and/or other elements.

It should be noted that Picture Renderer 91 and Picture Recognizer 92 can optionally be used to detect objects and/or their properties that cannot not be obtained from Application Program 18 or from an engine, environment, or system used to implement Application Program 18. Picture Renderer 91 and Picture Recognizer 92 can also optionally be used where Picture Renderer 91 and Picture Recognizer 92 offer superior performance in detecting objects and/or their properties. Picture Renderer 91 and Picture Recognizer 92 can also optionally be used to confirm objects and/or their properties obtained or detected by other means. For example, identity of an object, type of an object, and/or action of an object, if needed, can be recognized or confirmed through picture processing of Picture Renderer 91 and Picture Recognizer 92. Picture Renderer 91 and Picture Recognizer 92 can be omitted depending on implementation.

Referring to Fig. 4, an embodiment of utilizing Sound Renderer 96 and Sound Recognizer 97 is illustrated.

Sound Renderer 96 comprises the functionality for rendering or generating digital sound, and/or other functionalities. Sound Renderer 96 comprises the functionality for rendering or generating digital sound of Application Program 18. In some aspects, as a microphone is used to capture sound of a physical environment, Sound Renderer 96 can be used to render or generate sound of a computer modeled or represented environment. As such, Sound Renderer 96 can be used to render or generate sound of Application Program 18. In some designs, Sound Renderer 96 can be used to render or generate digital sound from Avatar's 605 surrounding in a 3D Application Program 18 (i.e. 3D computer game, virtual world application, CAD application, etc.). For example, emission of a sound from a sound source may be simulated/modeled in a virtual space of a 3D Application Program 18, propagation of the sound may be simulated/modeled through the virtual space including any scattering, reflections, refractions, diffractions, and/or other effects, and the sound may be rendered or generated as perceived by a listener (i.e. Avatar 605, etc.). In other designs, Sound Renderer 96 can be used to render or generate digital sound of a 2D Application Program 18 which may include any of the aforementioned and/or other sound simulation/modeling as applicable to 2D spaces. In further designs, Sound Renderer 96 can be optionally omitted in a simple Application Program 18 where no sound simulation/modeling is needed or where sounds may simply be played. In some implementations, Sound Renderer 96 may include any sound processing device, apparatus, system, or application that can render or generate digital sound. In some aspects, rendering, when used casually, may refer to rendering or generating digital sound from a computer model or representation, providing digital sound to a speaker or headphones, and/or producing the sound by a speaker or headphones. In some embodiments, Sound Renderer 96 can be a program executing or operating on Processor 11. In one example, Sound Renderer 96 can be provided in a rendering engine such as SoundScape Renderer, SLAB Spatial Audio Renderer, Uni-Verse Sound Renderer, Crepo Sound Renderer, and/or other programs or systems for rendering or processing sound. In another example, various engines or environments such as Unity 3D Engine, Unreal Engine, Torque 3D Engine,

and/or others provide built-in sound renderers. In other embodiments, Sound Renderer 96 can be part of, embedded into, or built into Processor 11. In further embodiments, Sound Renderer 96 can be a hardware element coupled to Processor 11 and/or other elements. In further embodiments, Sound Renderer 96 can be a program or hardware element that is part of or embedded into another element. In one example, a sound card and/or its processing unit may include Sound Renderer 96. In another example, ACAAO Unit 100 may include Sound Renderer 96. In a further example, Application Program 18 may include Sound Renderer 96. In general, Sound Renderer 96 can be implemented in any suitable configuration to provide its functionalities. Sound Renderer 96 may render or generate digital sound in various formats examples of which include WAV, WMA, AIFF, MP3, RA, OGG, and/or others. In some designs, Sound Renderer 96 can render or generate different digital sound of an Application Program 18 for production on a speaker or headphones and for facilitating object recognition functionalities herein. For example, sound of Avatar's 605 shooting may be produced by a speaker or headphones for User 50 to hear and sound of various objects in Avatar's 605 surrounding may be used to facilitate object recognition functionalities herein. In some implementations of non-acoustic Application Programs 18 such as simulations, calculations, and/or others, Sound Renderer 96 may render or generate digital sound as perceived by Avatar 605 to facilitate object recognition functionalities herein where the sound is never produced on a speaker or headphones. In some aspects, instead of or in addition to Sound Renderer 96, digital sound perceived by Avatar 605 can be obtained from any element of a computing device or system that can provide such digital sound. Examples of such elements include an audio circuit, an audio system, an audio driver, an audio interface, and/or others.

Sound Recognizer 97 comprises the functionality for detecting or recognizing objects and/or their properties in audio data, and/or other disclosed functionalities. Audio data includes digital sound, and/or other audio data. Examples of file formats that can be utilized to store audio data include WAV, WMA, AIFF, MP3, RA, OGG, and/or other file formats. In some designs, Sound Recognizer 97 can be used for detecting or recognizing objects and/or their properties in digital sound from Sound Renderer 96. In the case of a person, Sound Recognizer 97 may detect or recognize human voice. For example, Sound Recognizer 97 can be utilized in detecting or recognizing existence of an object, type of an object, identity of an object, activity of an object, and/or other properties of an object. In general, Sound Recognizer 97 can be used for any operation supported by Sound Recognizer 97. In some aspects, Sound Recognizer 97 may detect or recognize an object and/or its properties from a digital sound by comparing collections of sound samples from the digital sound with collections of sound samples of known objects and/or their properties. The collections of sound samples of known objects and/or their properties can be learned, or manually, programmatically, or otherwise defined. The collections of sound samples of known objects and/or their properties can be stored in any data structure or repository (i.e. one or more files, database, etc.) that resides locally on Computing Device 70, or remotely on a remote computing device (i.e. server, cloud, etc.) accessible over a network or an interface. In other aspects, Sound Recognizer 97 may detect or recognize an object and/or its properties from a digital sound by comparing features from the digital sound with features of sounds of known objects and/or their properties. The features of sounds of known objects and/or their properties can be learned, or manually, programmatically, or otherwise defined. The features of sounds of known objects and/or their properties can be stored in any data structure or repository (i.e. neural network, one or more files, database, etc.) that resides locally on Computing Device 70, or remotely on a remote computing device (i.e. server, cloud, etc.) accessible over a network or an interface. Typical steps or elements in a feature oriented sound recognition include pre-processing,

feature extraction, acoustic modeling, language modeling, and/or others, or a combination thereof, each of which may include its own sub-steps or sub-elements depending on the application. In further aspects, Sound Recognizer 97 may detect or recognize a variety of sounds from digital sound using the aforementioned sound sample or feature comparisons, and/or other detection or recognition techniques. For example, sound of a person, animal, vehicle, and/or other sounds can be detected by Sound Recognizer 97. In further aspects, Sound Recognizer 97 may detect or recognize sounds using Hidden Markov Models (HMM), Artificial Neural Networks, Dynamic Time Warping (DTW), Gaussian Mixture Models (GMM), and/or other models or techniques, or a combination thereof. Some or all of these models or techniques may include statistical techniques. Examples of artificial neural networks that can be used in Sound Recognizer 97 include recurrent neural networks, time delay neural networks (TDNNs), deep neural networks, convolutional neural networks, and/or others. In general, Sound Recognizer 97 may include any machine learning, deep learning, and/or other artificial intelligence techniques. Any other techniques known in art can be utilized in Sound Recognizer 97. For example, thresholds for similarity, statistical techniques, and/or optimization techniques can be utilized to determine a match in any of the above-described detection or recognition techniques.

In some exemplary embodiments, operating system's sound recognition functionalities such as iOS's Voice Services, Siri, and/or others can be utilized in Sound Recognizer 97. For example, iOS Voice Services can detect an object (i.e. person, etc.) and/or its properties in digital sound from Sound Renderer 96 or stored in an electronic repository, which can then be utilized in ACAAO Unit 100, Artificial Intelligence Unit 110, and/or other elements. In other exemplary embodiments, Java Speech API (JSAPI) implementation such as The Cloud Garden, Sphinx, and/or others can be utilized in Sound Recognizer 97. For example, Cloud Garden JSAPI can detect an object (i.e. person, animal, vehicle, etc.) and/or its properties in digital sound from Sound Renderer 96 or stored in an electronic repository, which can then be utilized in ACAAO Unit 100, Artificial Intelligence Unit 110, and/or other elements. Any other programming language's or platform's speech or sound processing API can similarly be utilized. In further exemplary embodiments, applications or engines providing Sound recognition functionalities such as HTK (Hidden Markov Model Toolkit), Kaldi, OpenEars, Dragon Mobile, Julius, iSpeech, CeedVocal, and/or others can be utilized in Sound Recognizer 97. For example, Kaldi SDK can detect an object (i.e. person, animal, vehicle, etc.) and/or its properties in digital sound from Sound Renderer 96 or stored in an electronic repository, which can then be utilized in ACAAO Unit 100, Artificial Intelligence Unit 110, and/or other elements.

It should be noted that Sound Renderer 96 and Sound Recognizer 97 can optionally be used to detect objects and/or their properties that cannot not be obtained from Application Program 18 or from an engine, environment, or system used to implement Application Program 18. Sound Renderer 96 and Sound Recognizer 97 can also optionally be used where Sound Renderer 96 and Sound Recognizer 97 offer superior performance in detecting objects and/or their properties. Sound Renderer 96 and Sound Recognizer 97 can also optionally be used to confirm objects and/or their properties obtained or detected by other means. For example, identity of an object, type of an object, and/or activity of an object, if needed, can be recognized or confirmed through sound processing of Sound Renderer 96 and Sound Recognizer 97. Sound Renderer 96 and Sound Recognizer 97 can be omitted depending on implementation.

One of ordinary skill in art will understand that the aforementioned techniques for detecting or recognizing objects and/or their properties using pictures and sounds are described merely as examples of a variety of possible

implementations, and that while all possible techniques for detecting or recognizing objects and/or their properties are too voluminous to describe, other techniques for detecting or recognizing objects and/or their properties known in art are within the scope of this disclosure. Also, any signal processing technique known in art that can facilitate the disclosed functionalities can be utilized in various embodiments. Any combination of the aforementioned and/or other renderers, object detecting or recognizing techniques, signal processing techniques, and/or other elements or techniques can be used in various embodiments.

Referring to Figs. 5A-5B, an exemplary embodiment of Objects 615 (also referred to simply as objects or other suitable name or reference) in Avatar's 605 surrounding, and resulting Collection of Object Representations 525 are illustrated.

As shown for example in Fig. 5A, Object 615a exists in Avatar's 605 surrounding. Object 615a may be recognized as a person. Object 615a may be located at a distance of 13m from Avatar 605. Object 615a may be located at a bearing/angle of 78° from Avatar's 605 centerline. Object 615a may be identified as Agent Smith. Furthermore, Object 615b exists in Avatar's 605 surrounding. Object 615b may be recognized as a rock. Object 615b may be located at a distance of 10m from Avatar 605. Object 615b may be located at a bearing/angle of 211° from Avatar's 605 centerline. Furthermore, Object 615c exists in Avatar's 605 surrounding. Object 615c may be recognized as a robot. Object 615c may be located at a distance of 8m from Avatar 605. Object 615c may be located at a bearing/angle of 332° from Avatar's 605 centerline. Any Objects 615 instead of or in addition to Object 615a, Object 615b, and Object 615c may exist in Avatar's 605 surrounding, one or more of which can be obtained, learned, and/or used. In some designs, some Objects 615 can be omitted. Which Objects 615 or types of Objects 615 are obtained, learned, and/or used can be defined by a user, by ACAAO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. In further designs, a 3D Application Program 18 may include elevated Objects 615 such as flying objects (i.e. flying animals, aircraft, etc.), objects on hills or mountains, objects on buildings, and/or others in which case altitudinal information related to distance and bearing/angle of Objects 615 relative to Avatar 605 can be obtained, learned, and/or used. Any unit of distance and/or bearing/angle can be utilized instead of or in addition to meters and/or angular degrees.

As shown for example in Fig. 5B, Object Processing Unit 140 may create or generate Collection of Object Representations 525 including Object Representation 625a representing Object 615a, Object Representation 625b representing Object 615b, Object Representation 625c representing Object 615c, etc. For instance, Object Representation 625a may include Object Property 630aa "Person" in Category 635aa "Type", Object Property 630ab "Agent Smith" in Category 635ab "Identity", Object Property 630ac "13m" in Category 635ac "Distance", Object Property 630ad "78°" in Category 635ad "Bearing", etc. Also, Object Representation 625b may include Object Property 630ba "Rock" in Category 635ba "Type", Object Property 630bb "10m" in Category 635bb "Distance", Object Property 630bc "211°" in Category 635bc "Bearing", etc. Also, Object Representation 625c may include Object Property 630ca "Robot" in Category 635ca "Type", Object Property 630cb "8m" in Category 635cb "Distance", Object Property 630cc "332°" in Category 635cc "Bearing", etc. Any number of Object Representations 625, and/or other elements or information can be included in Collection of Object Representations 525. Any number of Object Properties 630 (also referred to simply as object properties or other suitable name or reference), and/or other elements or information can be included in an Object Representation 625. In some aspects, a reference to

Collection of Object Representations 525 comprises a reference to a collection of Object Properties 630 and/or other elements or information related to one or more Objects 615. Other additional Object Representations 625, Object Properties 630, elements, and/or information can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate embodiments of Collection of Object Representations 525.

Referring now to ACAAO Unit 100, ACAAO Unit 100 comprises any hardware, programs, or a combination thereof. ACAAO Unit 100 comprises the functionality for learning the operation of Avatar 605 in circumstances including objects with various properties. ACAAO Unit 100 comprises the functionality for structuring and/or storing this knowledge in a knowledgebase (i.e. neural network, graph, sequences, other repository, etc.). ACAAO Unit 100 comprises the functionality for enabling autonomous operation of Avatar 605 in circumstances including objects with various properties. In some embodiments of Application Programs 18 that do not comprise Avatar 605 or rely on Avatar 605 for their operation, ACAAO Unit 100 comprises the functionality for learning the operation of Application Program 18 in circumstances including objects with various properties similar to the learning functionalities described with respect to Avatar 605. Also, in such embodiments, ACAAO Unit 100 comprises the functionality for enabling autonomous operation of Application Program 18 in circumstances including objects with various properties similar to the autonomous operation functionalities described with respect to Avatar 605. ACAAO Unit 100 comprises the functionality for interfacing with or attaching to Avatar 605, Application Program 18, Processor 11, and/or other processing element. ACAAO Unit 100 comprises the functionality for obtaining instruction sets, data, and/or other information used, implemented, and/or executed by Avatar 605, Application Program 18, Processor 11, and/or other processing element. ACAAO Unit 100 comprises the functionality for modifying instruction sets, data, and/or other information used, implemented, and/or executed by Avatar 605, Application Program 18, Processor 11, and/or other processing element. ACAAO Unit 100 comprises learning, anticipating, decision making, automation, and/or other functionalities disclosed herein. Statistical, artificial intelligence, machine learning, and/or other models or techniques are utilized to implement the disclosed devices, systems, and methods. In some designs, ACAAO Unit 100 and/or elements thereof may be or include a hardware element embedded or built into Processor 11, and/or other processing element. In other designs, ACAAO Unit 100 and/or elements thereof may be or include a hardware element coupled to or interfaced with Avatar 605, Application Program 18, Processor 11, and/or other processing element. In other designs, ACAAO Unit 100 and/or elements thereof may be or include a program operating on Processor 11, and/or other processing element. In further designs, ACAAO Unit 100 and/or elements thereof may be or include a program coupled to or interfaced with Avatar 605, Application Program 18, Processor 11, and/or other processing element. In further designs, ACAAO Unit 100 and/or elements thereof may be or include a program embedded or built into Avatar 605, Application Program 18, and/or other processing element. ACAAO Unit 100 can be provided in a combination of the aforementioned or other suitable configurations in alternate designs.

When ACAAO Unit 100 functionalities are applied to Avatar 605, Application Program 18, Processor 11, and/or other processing element, Avatar 605, Application Program 18, Processor 11, and/or other processing element may become autonomous. ACAAO Unit 100 may take control from, share control with, and/or release control to Avatar 605, Application Program 18, Processor 11, and/or other processing element to implement autonomous operation of Avatar 605, Application Program 18, Processor 11, and/or other processing element. ACAAO Unit 100 may take control from, share control with, and/or release control to Avatar 605, Application

Program 18, Processor 11, and/or other processing element automatically or after prompting User 50 to allow it. In some aspects, Avatar 605, Application Program 18, Processor 11, and/or other processing element may include or be provided with anticipatory (also referred to as alternate or other suitable name or reference) instructions or instruction sets that User 50 did not issue or cause to be executed. Such anticipatory instructions or instruction sets include instruction sets that User 50 may want or is likely to issue or cause to be executed. Anticipatory instructions or instruction sets can be generated by ACAAO Unit 100 or elements thereof based on circumstances including objects with various properties. As such, Avatar 605, Application Program 18, Processor 11, and/or other processing element may include or be provided with some or all original instructions or instruction sets and/or any anticipatory instructions or instruction sets generated by ACAAO Unit 100. Therefore, autonomous operating of Avatar 605, Application Program 18, Processor 11, and/or other processing element may include executing some or all original instructions or instruction sets and/or any anticipatory instructions or instruction sets generated by ACAAO Unit 100. In one example, ACAAO Unit 100 can overwrite or rewrite the original instructions or instruction sets with ACAAO Unit 100-generated instructions or instruction sets. In another example, ACAAO Unit 100 can insert or embed ACAAO Unit 100-generated instructions or instruction sets among the original instructions or instruction sets. In a further example, ACAAO Unit 100 can branch, redirect, or jump to ACAAO Unit 100-generated instructions or instruction sets from the original instructions or instruction sets.

In some embodiments, autonomous Avatar 605 operating comprises determining, by ACAAO Unit 100, a next instruction or instruction set to be executed based on Avatar's 605 circumstances including objects with various properties prior to the user issuing or causing to be executed the next instruction or instruction set. In other embodiments, autonomous Avatar 605 operating comprises determining, by ACAAO Unit 100, a next instruction or instruction set to be executed based on Avatar's 605 circumstances including objects with various properties prior to the system receiving the next instruction or instruction set.

In some embodiments, autonomous Avatar 605 operating includes a partially or fully autonomous operating. In an example involving partially autonomous Avatar 605 operating, a user confirms ACAAO Unit 100-generated instructions or instruction sets prior to their execution. In an example involving fully autonomous application operating, ACAAO Unit 100-generated instructions or instruction sets are executed without user or other system confirmation (i.e. automatically, etc.).

In some embodiments, a combination of ACAAO Unit 100 and other systems and/or techniques can be utilized to implement Avatar's 605 operation. In one example, ACAAO Unit 100 may be a primary or preferred system for implementing Avatar's 605 operation. While operating autonomously under the control of ACAAO Unit 100, Avatar 605 may encounter a circumstance including objects with various properties that has not been encountered or learned before. In such situations, User 50 and/or non-ACAAO system may take control of Avatar's 605 operation. ACAAO Unit 100 may take control again when Avatar 605 encounters a previously learned circumstance including objects with various properties. Naturally, ACAAO Unit 100 can learn Avatar's 605 operation in circumstances while User 50 and/or non-ACAAO system is in control of Avatar 605, thereby reducing or eliminating the need for future involvement of User 50 and/or non-ACAAO system. In another example, User 50 and/or non-ACAAO system may be a primary or preferred system for implementing Avatar's 605 operation. While operating under the control of User 50 and/or non-ACAAO system, User 50 and/or non-ACAAO system may release control to ACAAO Unit 100 for any reason (i.e. User 50 gets tired or distracted, non-ACAAO system gets stuck or

cannot make a decision, etc.), at which point Avatar 605 can be controlled by ACAAO Unit 100. In some designs, ACAAO Unit 100 may take control in certain special circumstances including objects with various properties where ACAAO Unit 100 may offer superior performance even though User 50 and/or non-ACAAO system may generally be preferred. Once Avatar 605 leaves such special circumstances, ACAAO Unit 100 may release control to User 50 and/or non-ACAAO system. In general, ACAAO Unit 100 can take control from, share control with, or release control to User 50, non-ACAAO system, and/or other system or process at any time, in any circumstances, and remain in control for any period of time as needed.

In some embodiments, ACAAO Unit 100 may control one or more elements of Avatar 605 while User 50 and/or non-ACAAO system may control other one or more elements of Avatar 605. For example, ACAAO Unit 100 may control Avatar's 605 movement, while User 50 and/or non-ACAAO system may control Avatar's 605 aiming and shooting. Any other combination of controlling various elements or functions of Avatar 605 by ACAAO Unit 100, User 50, and/or non-ACAAO system can be implemented.

In some embodiments, ACAAO Unit 100 enables learning of a particular User's 50 knowledge, methodology, or style of operating Avatar 605. In some aspects, learning of a particular User's 50 knowledge, methodology, or style of operating Avatar 605 includes learning the User's 50 directing or operating Avatar 605 in circumstances including objects with various properties. In one example, one User 50 may shoot an opponent while another User 50 may strike the opponent with a sword in a computer game. In another example, one User 50 may jump over an obstacle while another User 50 may move around the obstacle in a virtual world application. In a further example, one User 50 may drive fast while another User 50 may drive cautiously in a racing game, and so on. The knowledge of User's 50 methodology or style of operating Avatar 605 can be used to enable personalized autonomous operation of Avatar 605 specific to a particular User 50. Therefore, ACAAO-enabled Avatar 605 may exemplify User's 50 knowledge, methodology, or style of operating Avatar 605 as learned from User 50. In some designs, this functionality enables one or more ACAAO-enabled Avatars 605 to be utilized in Application Program 18 (i.e. computer game, virtual world, etc.) to assist User 50 in defeating an opponent or achieving another goal. For example, User 50 can utilize a team of ACAAO-enabled Avatars 605 each of which may exemplify User's 50 knowledge, methodology, or style of operating Avatar 605. In other designs, ACAAO Unit 100 enables a professional or other experienced Application Program 18 operator (i.e. game player, etc.) to record his/her knowledge, methodology, or style of operating Avatar 605 into Knowledgebase 530 (i.e. Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.) and/or other repository. User 50 can then sell or make available his/her knowledge, methodology, or style of operating Avatar 605 to other users who may want to implement User's 50 knowledge, methodology, or style of operating Avatar 605. Knowledgebase 530 and/or other repository comprising User's 50 knowledge, methodology, or style of operating Avatar 605 can be available to other users via a storage medium, via a network, or via other means. In some implementations, User's 50 knowledge, methodology, or style of operating Avatar 605 can be applied to or implemented on any Object 615 of Application Program 18 as applicable, thereby enabling any Object 615 to exemplify User's 50 knowledge, methodology, or style of operating as learned from User 50. For example, a computer game developer may associate Knowledgebase 530 comprising User's 50 knowledge, methodology, or style of operating Avatar 605 with an Object 615 (i.e. tank, robot, aircraft, etc.), thereby enabling the Object 615 to operate based on the knowledge in the Knowledgebase 530.

Referring now to Acquisition Interface 120, Acquisition Interface 120 comprises the functionality for obtaining and/or receiving instruction sets, data, and/or other information. Acquisition Interface 120 comprises the functionality for obtaining and/or receiving instruction sets, data, and/or other information related to the operation of Avatar 605, Application Program 18, Processor 11, and/or other processing element. Acquisition Interface 120 comprises the functionality for obtaining and/or receiving instruction sets, data, and/or other information from Avatar 605, Application Program 18, Processor 11, and/or other processing element. Acquisition Interface 120 comprises the functionality for obtaining and/or receiving instruction sets, data, and/or other information at runtime. In one example, Acquisition Interface 120 can obtain Instruction Sets 526 used or executed in operating Avatar 605 operation, and transmit the Instruction Sets 526 to Artificial Intelligence Unit 110 for learning Avatar's 605 operation in circumstances including objects with various properties. In another example, in Application Programs 18 that do not comprise Avatar 605 or do not rely on Avatar 605 for their operation, Acquisition Interface 120 can obtain Instruction Sets 526 used or executed in operating Application Program 18, and transmit the Instruction Sets 526 to Artificial Intelligence Unit 110 for learning Application Program's 18 operation in circumstances including objects with various properties. Acquisition Interface 120 also comprises the functionality for tracing or profiling of Avatar 605, Application Program 18, Processor 11, and/or other processing element. Tracing or profiling may include adding trace code or instrumentation to Avatar 605 (i.e. Avatar's 605 object code, etc.) or Application Program 18, and/or outputting trace information (i.e. instruction sets, data, and/or other information, etc.) to a receiving target. Acquisition Interface 120 further comprises the functionality for attaching to or interfacing with Avatar 605, Application Program 18, Processor 11, and/or other processing element. In some aspects, Acquisition Interface 120 can access and/or read runtime engine/environment, virtual machine, operating system, compiler, interpreter, translator, execution stack, file, object, data structure, and/or other computing system elements. In other aspects, Acquisition Interface 120 can access and/or read memory, storage, and/or other repository. In further aspects, Acquisition Interface 120 can access and/or read Processor 11 registers and/or other Processor 11 elements. In further aspects, Acquisition Interface 120 can access and/or read inputs and/or outputs of Avatar 605, Application Program 18, Processor 11, and/or other processing element. In further aspects, Acquisition Interface 120 can access and/or read functions, methods, procedures, routines, subroutines, and/or other elements of Avatar 605 and/or Application Program 18. In further aspects, Acquisition Interface 120 can access and/or read source code, bytecode, compiled, interpreted code, translated code, machine code, and/or other code. In further aspects, Acquisition Interface 120 can access and/or read values, variables, parameters, and/or other data or information. Acquisition Interface 120 also comprises the functionality for transmitting the obtained instruction sets, data, and/or other information to Artificial Intelligence Unit 110 and/or other elements. As such, Acquisition Interface 120 provides input into Artificial Intelligence Unit 110 for knowledge structuring, anticipating, decision making, and/or other functionalities later in the process. Acquisition Interface 120 also comprises other disclosed functionalities.

Acquisition Interface 120 can employ various techniques for obtaining instruction sets, data, and/or other information. In one example, Acquisition Interface 120 can attach to and/or obtain Avatar's 605, Application Program's 18, Processor's 11, and/or other processing element's instruction sets, data, and/or other information through tracing or profiling techniques. Tracing or profiling may be used for outputting Avatar's 605, Application Program's 18, Processor's 11, and/or other processing element's instruction sets, data, and/or other information at runtime. For instance, tracing or profiling may include adding trace code (i.e. instrumentation, etc.) to Avatar 605 (i.e.

Avatar's 605 object code, etc.) or Application Program 18, and/or outputting trace information to a specific target. The outputted trace information (i.e. instruction sets, data, and/or other information, etc.) can then be provided to or recorded into a file, data structure, repository, application, and/or other system or target that may receive such trace information. As such, Acquisition Interface 120 can utilize tracing or profiling to obtain instruction sets, data, and/or other information and provide them as input into Artificial Intelligence Unit 110. In some aspects, instrumentation can be performed in source code, bytecode, compiled code, interpreted code, translated code, machine code, and/or other code. In other aspects, instrumentation can be performed in various elements of a computing system such as memory, virtual machine, runtime engine/environment, operating system, compiler, interpreter, translator, processor registers, and/or other elements. In yet other aspects, instrumentation can be performed in various abstraction layers of a computing system such as in software layer (i.e. Application Program 18, etc.), in virtual machine (if VM is used), in operating system, in Processor 11, and/or in other layers or areas that may exist in a particular computing system implementation. In yet other aspects, instrumentation can be performed at various times in Avatar's 605 or Application Program's 18 execution such as source code write time, compile time, interpretation time, translation time, linking time, loading time, runtime, and/or others. In yet other aspects, instrumentation can be performed at various granularities or code segments such as some or all lines of code, some or all statements, some or all instructions or instruction sets, some or all basic blocks, some or all functions/routines/subroutines, and/or some or all other code segments.

In some embodiments, Avatar 605 (i.e. Avatar's 605 object code, etc.) or Application Program 18 can be automatically instrumented. For example, Acquisition Interface 120 can access Avatar's 605 or Application Program's 18 source code, bytecode, or machine code and select instrumentation points of interest. Selecting instrumentation points may include finding locations in the source code, bytecode, or machine code corresponding to function calls, function entries, function exits, object creations, object destructions, event handler calls, new lines (i.e. to instrument all lines of code, etc.), thread creations, throws, and/or other points of interest. Instrumentation code can then be inserted at the instrumentation points of interest to output Avatar's 605 or Application Program's 18 instruction sets, data, and/or other information. In response to executing instrumentation code, Avatar's 605 or Application Program's 18 instruction sets, data, and/or other information may be received by Acquisition Interface 120. In some aspects, Avatar's 605 or Application Program's 18 source code, bytecode, or machine code can be dynamically instrumented. For example, instrumentation code can be dynamically inserted into Avatar 605 (i.e. Avatar's 605 object code, etc.) or Application Program 18 at runtime.

In other embodiments, Avatar 605 or Application Program 18 can be manually instrumented. In one example, a programmer can instrument a function call by placing an instrumenting function such as traceAvatar(), traceApplication(), etc. immediately after the function call as in the following example.

```
Avatar.moveForward(12);
traceAvatar('Avatar.moveForward(12);');
```

In another example, an instrumenting function can be placed immediately before the function call, or at the beginning, end, or anywhere within the function itself. A programmer may instrument all function calls or only function calls of interest. In a further example, a programmer can instrument all lines of code or only code lines of interest. In a further example, a programmer can instrument other elements utilized or implemented within Avatar 605 or Application Program 18 such as objects and/or any of their functions, data structures and/or any of their

functions, event handlers and/or any of their functions, threads and/or any of their functions, and/or other elements or functions. Similar instrumentation as in the preceding examples can be performed automatically or dynamically. In some designs where manual code instrumentation is utilized, Acquisition Interface 120 can optionally be omitted and Avatar's 605 or Application Program's 18 instruction sets, data, and/or other information may be transmitted directly to Artificial Intelligence Unit 110.

In some embodiments, ACAAO Unit 100 can be selective in learning instruction sets, data, and/or other information to those implemented, utilized, or related to an object, data structure, repository, thread, function, and/or other element of Application Program 18. In some aspects, Acquisition Interface 120 can obtain Application Program's 18 instruction sets, data, and/or other information implemented, utilized, or related to a certain object in an object oriented Application Program 18. For example, Acquisition Interface 120 can obtain Application Program's 18 instruction sets, data, and/or other information implemented, utilized, or related to Avatar 605.

In some embodiments, various computing systems and/or platforms may provide native tools for obtaining instruction sets, data, and/or other information. Also, independent vendors may provide portable tools with similar functionalities that can be utilized across different computing systems and/or platforms. These native and portable tools may provide a wide range of functionalities such as instrumentation, tracing or profiling, logging application or system messages, outputting custom text messages, outputting objects or data structures, outputting functions/routines/subroutines or their invocations, outputting variable or parameter values, outputting thread or process behaviors, outputting call or other stacks, outputting processor registers, providing runtime memory access, providing inputs and/or outputs, performing live application monitoring, and/or other capabilities. One of ordinary skill in art will understand that, while all possible variations of the techniques to obtain instruction sets, data, and/or other information are too voluminous to describe, these techniques are within the scope of this disclosure.

In one example, obtaining Avatar's 605 and/or Application Program's 18 instruction sets, data, and/or other information can be implemented through the .NET platform's native tools for application tracing or profiling such as System.Diagnostics.Trace, System.Diagnostics.Debug, or System.Diagnostics.TraceSource classes for tracing execution flow, and/or System.Diagnostics.Process, System.Diagnostics.EventLog, or System.Diagnostics.PerformanceCounter classes for profiling code, accessing local and remote processes, starting and stopping system processes, and interacting with Windows event logs, etc. For instance, a set of trace switches can be created that output an application's information. The switches can be configured using the .config file. For a web application, this may typically be web.config file associated with the project. In a Windows application, this file may typically be named applicationName.exe.config. Trace code can be added to application code automatically or manually as previously described. Appropriate listener can be created where the trace output is received. Trace code can output trace messages to a specific target such as a file, a log, a database, an object, a data structure, and/or other repository or system. Acquisition Interface 120 or Artificial Intelligence Unit 110 can then read or obtain the trace information from these targets. In some aspects, trace code can output trace messages directly to Acquisition Interface 120. In other aspects, trace code can output trace messages directly to Artificial Intelligence Unit 110. In the case of outputting trace messages to Acquisition Interface 120 or directly to Artificial Intelligence Unit 110, custom listeners can be built to accommodate these specific targets. Other platforms, tools, and/or techniques can provide equivalent or similar functionalities as the above described ones.

In another example, obtaining Avatar's 605 and/or Application Program's 18 instruction sets, data, and/or

other information can be implemented through the .NET platform's Profiling API that can be used to create a custom profiler application for tracing, monitoring, interfacing with, and/or managing a profiled application. The Profiling API provides an interface that includes methods to notify the profiler of events in the profiled application. The Profiling API may also provide an interface to enable the profiler to call back into the profiled application to obtain information about the state of the profiled application. The Profiling API may further provide call stack profiling functionalities. Call stack (also referred to as execution stack, control stack, runtime stack, machine stack, the stack, etc.) includes a data structure that can store information about active subroutines of an application. The Profiling API may provide a stack snapshot method, which enables a trace of the stack at a particular point in time. The Profiling API may also provide a shadow stack method, which tracks the call stack at every instant. A shadow stack can obtain function arguments, return values, and information about generic instantiations. A function such as FunctionEnter can be utilized to notify the profiler that control is being passed to a function and can provide information about the stack frame and function arguments. A function such as FunctionLeave can be utilized to notify the profiler that a function is about to return to the caller and can provide information about the stack frame and function return value. An alternative to call stack profiling includes call stack sampling in which the profiler can periodically examine the stack. In some aspects, the Profiling API enables the profiler to change the in-memory code stream for a routine before it is just-in-time (JIT) compiled where the profiler can dynamically add instrumentation code to all or particular routines of interest. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining Avatar's 605 and/or Application Program's 18 instruction sets, data, and/or other information can be implemented through Java platform's APIs for application tracing or profiling such as Java Virtual Machine Profiling Interface (JVMPI), Java Virtual Machine Tool Interface (JVMTI), and/or other APIs or tools. These APIs can be used for instrumentation of an application, for notification of Java Virtual Machine (VM) events, and/or other functionalities. One of the tracing or profiling techniques that can be utilized includes bytecode instrumentation. The profiler can insert bytecodes into all or some of the classes. In application execution profiling, for example, these bytecodes may include methodEntry and methodExit calls. In memory profiling, for example, the bytecodes may be inserted after each new or after each constructor. In some aspects, insertion of instrumentation bytecode can be performed either by a post-compiler or a custom class loader. An alternative to bytecode instrumentation includes monitoring events generated by the JVMPI or JVMTI interfaces. Both APIs can generate events for method entry/exit, object allocation, and/or other events. In some aspects, JVMTI can be utilized for dynamic bytecode instrumentation where insertion of instrumentation bytecodes is performed at runtime. The profiler may insert the necessary instrumentation when a selected class is invoked in an application. This can be accomplished using the JVMTI's redefineClasses method, for example. This approach also enables changing of the level of profiling as the application is running. If needed, these changes can be made adaptively without restarting the application. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining Avatar's 605 and/or Application Program's 18 instruction sets, data, and/or other information can be implemented through JVMTI's programming interface that enables creation of software agents that can monitor and control a Java application. An agent may use the functionality of the interface to register for notification of events as they occur in the application, and to query and control the application. A JVMTI agent

may use JVMTI functions to extract information from a Java application. A JVMTI agent can be utilized to obtain an application's runtime information such as method calls, memory allocation, CPU utilization, and/or other information. JVMTI may include functions to obtain information about variables, fields, methods, classes, and/or other information. JVMTI may also provide notification for numerous events such as method entry and exit, exception, field access and modification, thread start and end, and/or other events. Examples of JVMTI built-in methods include GetMethodName to obtain the name of an invoked method, GetThreadInfo to obtain information for a specific thread, GetClassSignature to obtain information about the class of an object, GetStackTrace to obtain information about the stack including information about stack frames, and/or other methods. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining Avatar's 605 and/or Application Program's 18 instruction sets, data, and/or other information can be implemented through java.lang.Runtime class that provides an interface for application tracing or profiling. Examples of methods provided in java.lang.Runtime that can be used to obtain an application's instruction sets, data, and/or other information include tracemethodcalls, traceinstructions, and/or other methods. These methods prompt the Java Virtual Machine to output trace information for a method or instruction in the virtual machine as it is executed. The destination of trace output may be system dependent and include a file, a listener, and/or other destinations where Acquisition Interface 120, Artificial Intelligence Unit 110, and/or other disclosed elements can access needed information. In addition to tracing or profiling tools native to their respective computing systems and/or platforms, many independent tools exist that provide tracing or profiling functionalities on more than one computing system and/or platform. Examples of these tools include Pin, DynamoRIO, KernInst, DynInst, Kprobes, OpenPAT, DTrace, SystemTap, and/or others. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining Avatar's 605 and/or Application Program's 18 instruction sets, data, and/or other information can be implemented through logging tools of the platform and/or operating system on which an application runs. Some logging tools may include nearly full feature sets of the tracing or profiling tools previously described. In one example, Visual Basic enables logging of runtime messages through its Microsoft.VisualBasic.Logging namespace that provides a log listener where the log listener may direct logging output to a file and/or other target. In another example, Java enables logging through its java.util.logging class. In some aspects, obtaining an application's instruction sets, data, and/or other information can be implemented through logging capabilities of the operating system on which an application runs. For example, Windows NT features centralized log service that applications and operating-system components can utilize to report their events including any messages. Windows NT provides functionalities for system, application, security, and/or other logging. An application log may include events logged by applications. Windows NT, for example, may include support for defining an event source (i.e. application that created the event, etc.). Windows Vista, for example, supports a structured XML log-format and designated log types to allow applications to more precisely log events and to help interpret the events. Examples of different types of event logs include administrative, operational, analytic, debug, and/or other log types including any of their subcategories. Examples of event attributes that can be utilized include eventID, level, task, opcode, keywords, and/or other event attributes. Windows wevtutil tool enables access to events, their structures, registered event publishers, and/or their configuration even before the events are fired. Wevtutil supports capabilities such as retrieval of the names of all logs on a computing device; retrieval of

configuration information for a specific log; retrieval of event publishers on a computing device; reading events from an event log, from a log file, or using a structured query; exporting events from an event log, from a log file, or using a structured query to a specific target; and/or other capabilities. Operating system logs can be utilized solely if they contain sufficient information on an application's instruction sets, data, and/or other information. Alternatively, operating system logs can be utilized in combination with another source of information (i.e. trace information, call stack, processor registers, memory, etc.) to reconstruct the application's instruction sets, data, and/or other information needed for Artificial Intelligence Unit 110 and/or other elements. In addition to logging capabilities native to their respective platforms and/or operating systems, many independent tools exist that provide logging on different platforms and/or operating systems. Examples of these tools include Log4j, Logback, SmartInspect, NLog, log4net, Microsoft Enterprise Library, ObjectGuy Framework, and/or others. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining Avatar's 605 and/or Application Program's 18 instruction sets, data, and/or other information can be implemented through tracing or profiling the operating system on which an application runs. As in tracing or profiling an application, one of the techniques that can be utilized includes adding instrumentation code to the operating system's source code. Such instrumentation code can be added to the operating system's source code before kernel compilation or recompilation, for instance. This type of instrumentation may involve defining or finding locations in the operating system's source code where instrumentation code may be inserted. Kernel instrumentation can also be performed without the need for kernel recompilation or rebooting. In some aspects, instrumentation code can be added at locations of interest through binary rewriting of compiled kernel code. In other aspects, kernel instrumentation can be performed dynamically where instrumentation code is added and/or removed where needed at runtime. For instance, dynamic instrumentation may overwrite kernel code with a branch instruction that redirects execution to instrumentation code or instrumentation routine. In yet other aspects, kernel instrumentation can be performed using just-in-time (JIT) dynamic instrumentation where execution may be redirected to a copy of kernel's code segment that includes instrumentation code. This type of instrumentation may include a JIT compiler and creation of a copy of the original code segment having instrumentation code or calls to instrumentation routines embedded into the original code segment. Instrumentation of the operating system may enable total system visibility including visibility into an application's behavior by enabling generation of low level trace information. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining Avatar's 605 and/or Application Program's 18 instruction sets, data, and/or other information can be implemented through tracing or profiling the processor on which an application runs. For example, some Intel processors provide Intel Processor Trace (i.e. Intel PT, etc.), a low-level tracing feature that enables recording executed instruction sets, and/or other data or information of one or more applications. Intel PT is facilitated by the Processor Trace Decoder Library along with its related tools. Intel PT is a low-overhead execution tracing feature that records information about application execution on each hardware thread using dedicated hardware facilities. The recorded execution/trace information is collected in data packets that can be buffered internally before being sent to a memory subsystem or another system or element (i.e. Acquisition Interface 120, Artificial Intelligence Unit 110, etc.). Intel PT also enables navigating the recorded execution/trace information via reverse stepping commands. Intel PT can be included in an operating system's core files and provided as a feature

of the operating system. Intel PT can trace globally some or all applications running on an operating system. Acquisition Interface 120 or Artificial Intelligence Unit 110 can read or obtain the recorded execution/trace information from Intel PT. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining Avatar's 605 and/or Application Program's 18 instruction sets, data, and/or other information can be implemented through branch tracing or profiling. Branch tracing may include an abbreviated instruction trace in which only the successful branch instruction sets are traced or recorded. Branch tracing can be implemented through utilizing dedicated processor commands, for example. Executed branches may be saved into special branch trace store area of memory. With the availability and reference to a compiler listing of the application together with branch trace information, a full path of executed instruction sets can be reconstructed. The full path can also be reconstructed with a memory dump (containing the program storage) and branch trace information. In some aspects, branch tracing can be utilized for pre-learning or automated learning of an application's instruction sets, data, and/or other information where a number of application simulations (i.e. simulations of likely/common operations, etc.) are performed. As such, the application's operation can be learned automatically saving the time that would be needed to learn the application's operation directed by a user. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining Avatar's 605 and/or Application Program's 18 instruction sets, data, and/or other information can be implemented through assembly language. Assembly language is a low-level programming language for a computer or other programmable device in which there is a strong correlation between the language and the architecture's machine instruction sets. Syntax, addressing modes, operands, and/or other elements of an assembly language instruction set may translate directly into numeric (i.e. binary, etc.) representations of that particular instruction set. Because of this direct relationship with the architecture's machine instruction sets, assembly language can be a powerful tool for tracing or profiling an application's execution in processor registers, memory, and/or other computing system components. For example, using assembly language, memory locations of a loaded application can be accessed, instrumented, and/or otherwise manipulated. In some aspects, assembly language can be used to rewrite or overwrite original in-memory instruction sets of an application with instrumentation instruction sets. In other aspects, assembly language can be used to redirect application's execution to instrumentation routine/subroutine or other code segment elsewhere in memory by inserting a jump into the application's in-memory code, by redirecting program counter, or by other techniques. Some operating systems may implement protection from changes to applications loaded into memory. Operating system, processor, or other low level commands such as Linux mprotect command or similar commands in other operating systems may be used to unprotect the protected locations in memory before the change. In yet other aspects, assembly language can be used to obtain instruction sets, data, and/or other information through accessing and/or reading instruction register, program counter, other processor registers, memory locations, and/or other components of a computing system. In yet other aspects, high-level programming languages may call or execute an external assembly language program to facilitate obtaining instruction sets, data, and/or other information as previously described. In yet other aspects, relatively low-level programming languages such as C may allow embedding assembly language directly in their source code such as, for example, using asm keyword of C. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, it may be sufficient to obtain user or other inputs, variables, parameters, and/or other data in some procedural, simple object oriented, or other applications. In one instance, a simple procedural application executes a sequence of instruction sets until the end of the program. During its execution, the application may receive user or other input, store the input in a variable, and perform calculations using the variable to reach a result. The value of the variable can be obtained or traced. In another instance, a more complex procedural application comprises one or more functions/routines/subroutines each of which may include a sequence of instruction sets. The application may execute a main sequence of instruction sets with a branch to a function/routine/subroutine. During its execution, the application may receive user or other input, store the input in a variable, and pass the variable as a parameter to the function/routine/subroutine. The function/routine/subroutine may perform calculations using the parameter and return a value that the rest of the application can use to reach a result. The value of the variable or parameter passed to the function/routine/subroutine, and/or return value can be obtained or traced. Values of user or other inputs, variables, parameters, and/or other items of interest can be obtained through previously described tracing, instrumentation, and/or other techniques. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

Referring to Fig. 6, in yet another example, obtaining Avatar's 605 and/or Application Program's 18 instruction sets, data, and/or other information may be implemented through tracing, profiling, or sampling of instruction sets or data in processor registers, memory, or other computing system components where instruction sets, data, and/or other information may be stored or utilized. For example, Instruction Register 212 may be part of Processor 11 and it may store the instruction set currently being executed or decoded. In some processors, Program Counter 211 (also referred to as instruction pointer, instruction address register, instruction counter, or part of instruction sequencer, etc.) may be incremented after fetching an instruction set, and it may hold or point to the memory address of the next instruction set to be executed. In a processor where the incrementation precedes the fetch, Program Counter 211 may point to the current instruction set being executed. In the instruction cycle, an instruction set may be loaded into Instruction Register 212 after Processor 11 fetches it from location in Memory 12 pointed to by Program Counter 211. Instruction Register 212 may hold the instruction set while it is decoded by Instruction Decoder 213, prepared, and executed. In some aspects, data (i.e. operands, etc.) needed for instruction set execution may be loaded from Memory 12 into a register within Register Array 214. In other aspects, the data may be loaded directly into Arithmetic Logic Unit 215. For instance, as instruction sets pass through Instruction Register 212 during application execution, they may be transmitted to Acquisition Interface 120 as shown. Examples of the steps in execution of a machine instruction set may include decoding the opcode (i.e. portion of a machine instruction set that may specify the operation to be performed), determining where the operands may be located (depending on architecture, operands may be in registers, the stack, memory, I/O ports, etc.), retrieving the operands, allocating processor resources to execute the instruction set (needed in some types of processors), performing the operation indicated by the instruction set, saving the results of execution, and/or other execution steps. Examples of the types of machine instruction sets that can be utilized include arithmetic, data handling, logical, program control, as well as special and/or other instruction set types. In addition to the ones described or shown, examples of other computing system or processor components that can be used during an instruction cycle include memory address register (MAR) that may hold the address of a memory block to be read from or written to; memory data register (MDR) that may hold data fetched from memory or data waiting to be stored in memory; data

registers that may hold numeric values, characters, small bit arrays, or other data; address registers that may hold addresses used by instruction sets that indirectly access memory; general purpose registers (GPRs) that may store both data and addresses; conditional registers that may hold truth values often used to determine whether some instruction set should or should not be executed; floating point registers (FPRs) that may store floating point numbers; constant registers that may hold read-only values such as zero, one, or pi; special purpose registers (SPRs) such as status register, program counter, or stack pointer that may hold information on program state; machine-specific registers that may store data and settings related to a particular processor; Register Array 214 that may include an array of any number of processor registers; Arithmetic Logic Unit 215 that may perform arithmetic and logic operations; control unit that may direct processor's operation; and/or other circuits or components. Tracing, profiling, or sampling of processor registers, memory, or other computing system components can be implemented in a program, combination of hardware and program, or purely hardware system. Dedicated hardware may be built to perform tracing, profiling, or sampling of processor registers or any computing system components with marginal or no impact to computing overhead.

One of ordinary skill in art will recognize that Fig. 6 depicts one of many implementations of processor or computing system components, and that various additional components can be included, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate implementations. Processor or computing system components may be arranged or connected differently in alternate implementations. Processor or computing system components may also be connected with external elements using various connections. For instance, the connection between Instruction Register 212 and Acquisition Interface 120 may include any number or types of connections such as, for example, a dedicated connection for each bit of Instruction Register 212 (i.e. 32 connections for a 32 bit Instruction Register 212, etc.). Any of the described or other connections or interfaces may be implemented among any processor or computing system components and Acquisition Interface 120 or other elements.

Other additional techniques or elements can be utilized as needed for obtaining instruction sets, data, and/or other information, or some of the disclosed techniques or elements can be excluded, or a combination thereof can be utilized in alternate embodiments. As an avatar (i.e. Avatar 605, etc.) may be part of an application (i.e. Application Program 18, etc.), it should be noted that obtaining an avatar's instruction sets, data, and/or other information may include same or similar techniques as the aforementioned obtaining an application's instruction sets, data, and/or other information, and vice versa.

Referring to Figs. 7A-7E, some embodiments of Instruction Sets 526 are illustrated. In some aspects, Instruction Set 526 includes one or more instructions or commands related to Avatar 605. For example, Instruction Set 526 may include one or more instructions or commands for operating Avatar 605. In other aspects, Instruction Set 526 includes one or more instructions or commands of Application Program 18. For example, Instruction Set 526 may include one or more instructions or commands for operating Application Program 18. In further aspects, Instruction Set 526 includes one or more inputs into and/or outputs from Avatar 605, Application Program 18, Processor 11, and/or other processing element. In further aspects, Instruction Set 526 includes one or more values or states of registers and/or other components of Processor 11 and/or other processing element. In general, Instruction Set 526 may include one or more instructions, commands, keywords, symbols (i.e. parentheses, brackets, commas, semicolons, etc.), operators (i.e. =, <, >, etc.), variables, values, objects, data structures,

functions (i.e. Function1(), FIRST(), MIN(), SQRT(), etc.), parameters, states, signals, inputs, outputs, characters, digits, references thereto, and/or other components. Therefore, the terms instruction set, command, instruction, signal, or other such terms may be used interchangeably herein depending on context.

In an embodiment shown in Fig. 7A, Instruction Set 526 includes code of a high-level programming language (i.e. Java, C++, etc.) comprising the following function call construct: Function1 (Parameter1, Parameter2, Parameter3, ...). An example of a function call applying the above construct includes the following Instruction Set 526: moveTo(Avatar, 11, 23). The function or reference thereto "moveTo(Avatar, 11, 23)" may be an Instruction Set 526 directing Avatar 605 to move to a location with coordinates 11 and 23, for example. In another embodiment shown in Fig. 7B, Instruction Set 526 includes structured query language (SQL). In a further embodiment shown in Fig. 7C, Instruction Set 526 includes bytecode (i.e. Java bytecode, Python bytecode, CLR bytecode, etc.). In a further embodiment shown in Fig. 7D, Instruction Set 526 includes assembly code. In a further embodiment shown in Fig. 7E, Instruction Set 526 includes machine code. Instruction Set 526 may include any other language or construct in alternate embodiments.

Referring to Figs. 8A-8B, some embodiments of Extra Information 527 (also referred to as Extra Info 527) are illustrated. In an embodiment shown in Fig. 8A, Collection of Object Representations 525 may include or be associated with Extra Info 527. In an embodiment shown in Fig. 8B, Instruction Set 526 may include or be associated with Extra Info 527.

Extra Info 527 comprises the functionality for storing any information useful in comparisons or decision making performed in autonomous Avatar 605 operation, and/or other functionalities. One or more Extra Infos 527 can be stored in, appended to, or associated with a Collection of Object Representations 525, Instruction Set 526, and/or other element. In some embodiments, the system can obtain Extra Info 527 at a time of creating or generating Collection of Object Representations 525. In other embodiments, the system can obtain Extra Info 527 at a time of acquiring Instruction Set 526. In general, Extra Info 527 can be obtained at any time. Examples of Extra Info 527 include time information, location information, computed information, visual information, acoustic information, contextual information, and/or other information. Any information can be utilized that can provide information for enhanced comparisons or decision making performed in autonomous Avatar 605 operation. Which information is utilized and/or stored in Extra Info 527 can be set by a user, by ACAAO system administrator, or automatically by the system. Extra Info 527 may include or be referred to as contextual information, and vice versa. Therefore, these terms may be used interchangeably herein depending on context.

In some aspects, time information (i.e. time stamp, etc.) can be utilized and/or stored in Extra Info 527. Time information can be useful in comparisons or decision making performed in autonomous Avatar 605 operation related to a specific time period. For example, Extra Info 527 may include time information related to when Avatar 605 performed an operation. Time information can be obtained from the system clock, online clock, oscillator, or other time source. In other aspects, location information (i.e. coordinates, etc.) can be utilized and/or stored in Extra Info 527. Location information can be useful in comparisons or decision making performed in autonomous Avatar 605 operation related to a specific place. For example, Extra Info 527 may include location information related to where Avatar 605 performed an operation. Location information can be obtained from Application Program's 18 engine (i.e. game engine in which the game is implemented, etc.), runtime environment, functions for providing location information on objects, and/or others as previously described. In further aspects, computed information can

be utilized and/or stored in Extra Info 527. Computed information can be useful in comparisons or decision making performed in autonomous Avatar 605 operation where information can be calculated, inferred, or estimated from other available information. ACAAO Unit 100 may include computational functionalities to create Extra Info 527 by performing calculations, inferences, or estimations using other information. In one example, Avatar's 605 direction of movement can be computed or estimated using Avatar's 605 location information. In another example, Avatar's 605 speed can be computed or estimated using Avatar's 605 location and/or time information. In a further example, speeds, directions of movement, trajectories, distances, and/or other properties of objects around Avatar 605 can similarly be computed or estimated, thereby providing geo-spatial and situational awareness and/or capabilities to Avatar 605. ACAAO Unit 100 can utilize geometry, trigonometry, Pythagorean theorem, and/or other theorems, formulas, or disciplines in its calculations, inferences, or estimations. In further aspects, visual information can be utilized and/or stored in Extra Info 527. Visual information can be useful in comparisons or decision making performed in autonomous Avatar 605 operation related to an object or environment that can be recognized from visual information. For example, an object or environment can be recognized by processing one or more digital pictures from Picture Renderer 91, visual processor, visual program, or other visual provider. Any features, functionalities, and embodiments of Picture Recognizer 92 can be utilized for such recognizing as previously described. For instance, trees recognized in the background of one or more digital pictures from Picture Renderer 91 may indicate a park or forest. In further aspects, acoustic information can be utilized and/or stored in Extra Info 527. Acoustic information can be useful in comparisons or decision making performed in autonomous Avatar 605 operation related to a sound or accosting environment. For example, an object or environment can be recognized by processing digital sound from Sound Renderer 96, sound processor, sound program, or other sound provider. Any features, functionalities, and embodiments of Sound Recognizer 97 can be utilized for such recognizing as previously described. For instance, sound of a horn recognized in digital sound from Sound Renderer 96 may indicate a proximal vehicle. In further aspects, other information can be utilized and/or stored in Extra Info 527. Examples of such other information include user specific information (i.e. skill level, age, gender, etc.), group user information (i.e. access level, etc.), version of Application Program 18, type of Application Program 18, type of Avatar 605, name of Avatar 605, allegiance of Avatar 605, type of Processor 11, type of Computing Device 70, and/or other information all of which can be obtained from various devices, systems, repositories, functions, or elements of Computing Device 70, Processor 11, Application Program 18, Avatar 605, and/or other processing elements.

Referring to Fig. 9, an embodiment where ACAAO Unit 100 is part of or operating on Processor 11 is illustrated. In one example, ACAAO Unit 100 may be a hardware element or circuit embedded or built into Processor 11. In another example, ACAAO Unit 100 may be a program operating on Processor 11.

Referring to Fig. 10, an embodiment where ACAAO Unit 100 resides on Server 96 accessible over Network 95 is illustrated. Any number of Computing Devices 70, Processors 11, Application Programs 18, and/or other elements may connect to such remote ACAAO Unit 100 and the remote ACAAO Unit 100 may learn operations of their Avatars 605 in circumstances including objects with various properties. In turn, any number of Computing Devices 70, Processors 11, Application Programs 18, and/or other elements can utilize the remote ACAAO Unit 100 for autonomous operation of their Avatars 605. A remote ACAAO Unit 100 can be offered as a network service (i.e. online application, etc.). In some aspects, a remote ACAAO Unit 100 (i.e. global ACAAO Unit 100, etc.) may reside

on the Internet and be available to all the world's Computing Devices 70, Processors 11, Application Programs 18, and/or other elements configured to transmit operations of their Avatars 605 in circumstances including objects with various properties and/or configured to utilize the remote ACAAO Unit 100 for autonomous operation of their Avatars 605. For example, multiple players (i.e. Users 50, etc.) may operate their Avatars 605 in a computer game (i.e. Application Program 18, etc.) running on their respective Computing Devices 70 where the Computing Devices 70 and/or elements thereof may be configured to transmit Avatar's 605 operations in circumstances including objects with various properties to a remote ACAAO Unit 100. Such remote ACAAO Unit 100 enables learning of the players' collective knowledge of operating Avatar 605 in circumstances including objects with various properties. Server 96 may be or include any type or form of a remote computing device such as an application server, a network service server, a cloud server, a cloud, and/or other remote computing device. Server 96 may include any features, functionalities, and embodiments of Computing Device 70. It should be understood that Server 96 does not have to be a separate computing device and that Server 96, its elements, or its functionalities can be implemented on Computing Device 70. Network 95 may include various networks, connection types, protocols, interfaces, APIs, and/or other elements or techniques known in art all of which are within the scope of this disclosure. Any of the previously described networks, network or connection types, networking interfaces, and/or other networking elements or techniques can similarly be utilized. Any of the disclosed elements can reside on Server 96 in alternate implementations. In one example, Artificial Intelligence Unit 110 can reside on Server 96 and Acquisition Interface 120, Modification Interface 130, and/or Object Processing Unit 140 can reside on Computing Device 70. In another example, Knowledgebase 530 (later described) can reside on Server 96 and the rest of the elements of ACAAO Unit 100 can reside on Computing Device 70. Any other combination of local and remote elements can be implemented.

Referring to Fig. 11, an embodiment of Artificial Intelligence Unit 110 is illustrated. Artificial Intelligence Unit 110 comprises interconnected Knowledge Structuring Unit 520, Knowledgebase 530, Decision-making Unit 540, and Confirmation Unit 550. Other additional elements can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate embodiments.

Artificial Intelligence Unit 110 comprises the functionality for learning Avatar's 605 operation in circumstances including objects with various properties. Artificial Intelligence Unit 110 comprises the functionality for learning one or more collections of object representations correlated with any instruction sets, data, and/or other information. In some aspects, Artificial Intelligence Unit 110 comprises the functionality for learning one or more Collections of Object Representations 525 representing Objects 615 in Avatar's 605 surrounding correlated with any Instruction Sets 526 and/or Extra Info 527. The Instruction Sets 526 may be used or executed in operating Avatar 605. In other aspects, Artificial Intelligence Unit 110 comprises the functionality for learning one or more Collections of Object Representations 525 representing Objects 615 in Avatar's 605 surrounding some of which may not be correlated with any Instruction Sets 526 and/or Extra Info 527. Further, Artificial Intelligence Unit 110 comprises the functionality for anticipating Avatar's 605 operation in circumstances including objects with various properties. Artificial Intelligence Unit 110 comprises the functionality for anticipating one or more instruction sets, data, and/or other information. Artificial Intelligence Unit 110 comprises the functionality for anticipating one or more Instruction Sets 526 (i.e. anticipatory Instruction Sets 526, etc.) based on one or more incoming Collections of Object Representations 525 representing Objects 615 in Avatar's 605 surrounding. The one or more Instruction Sets 526 (i.e. anticipatory Instruction Sets 526, etc.) may be used or executed in Avatar's 605 autonomous operation. In some

embodiments of Application Programs 18 that do not comprise Avatar 605 or rely on Avatar 605 for their operation, Artificial Intelligence Unit 110 comprises the functionality for learning Application Program's 18 operation in circumstances including objects with various properties similar to the learning functionalities described with respect to Avatar 605. Also, in such embodiments, Artificial Intelligence Unit 110 comprises the functionality for anticipating Application Program's 18 operation in circumstances including objects with various properties similar to the anticipating functionalities described with respect to Avatar 605. Artificial Intelligence Unit 110 also comprises other disclosed functionalities.

Knowledge Structuring Unit 520, Knowledgebase 530, and Decision-making Unit 540 are described later.

Confirmation Unit 550 comprises the functionality for confirming, modifying, evaluating (i.e. rating, etc.), and/or canceling one or more anticipatory Instruction Sets 526, and/or other functionalities. Confirmation Unit 550 is an optional element that can be omitted depending on implementation. In some embodiments, Confirmation Unit 550 can serve as a means of confirming anticipatory Instruction Sets 526. For example, Decision-making Unit 540 may determine one or more anticipatory Instruction Sets 526 and provide them to User 50 for confirmation. User 50 may be provided with an interface (i.e. graphical user interface, selectable list of anticipatory Instruction Sets 526, etc.) to approve or confirm execution of the anticipatory Instruction Sets 526. In some aspects, Confirmation Unit 550 can automate User 50 confirmation. In one example, if one or more incoming Collections of Object Representations 525 from Object Processing Unit 140 and one or more Collections of Object Representations 525 from a Knowledge Cell 800 were found to be a perfect or highly similar match, anticipatory Instruction Sets 526 correlated with the one or more Collections of Object Representations 525 from the Knowledge Cell 800 can be automatically executed without User's 50 confirmation. Conversely, if one or more incoming Collections of Object Representations 525 from Object Processing Unit 140 and one or more Collections of Object Representations 525 from a Knowledge Cell 800 were found to be less than a highly similar match, anticipatory Instruction Sets 526 correlated with the one or more Collections of Object Representations 525 from the Knowledge Cell 800 can be presented to User 50 for confirmation and/or modifying. Any features, functionalities, and/or embodiments of Similarity Comparison 125 (later described) can be utilized for such similarity determination. In other embodiments, Confirmation Unit 550 can serve as a means of modifying or editing anticipatory Instruction Sets 526. For example, Decision-making Unit 540 may determine one or more anticipatory Instruction Sets 526 and provide them to User 50 for modification. User 50 may be provided with an interface (i.e. graphical user interface, etc.) to modify the anticipatory Instruction Sets 526 before their execution. In further embodiments, Confirmation Unit 550 can serve as a means of evaluating or rating anticipatory Instruction Sets 526 if they matched User's 50 intended operation. For example, Decision-making Unit 540 may determine one or more anticipatory Instruction Sets 526, which the system may automatically execute. User 50 may be provided with an interface (i.e. graphical user interface, etc.) to rate (i.e. on a scale from 0 to 1, etc.) how well Decision-making Unit 540 predicted the executed anticipatory Instruction Sets 526. In some aspects, rating can be automatic and based on a particular function or method that rates how well the anticipatory Instruction Sets 526 matched the desired operation. In one example, a rating function or method can assign a higher rating to anticipatory Instruction Sets 526 that were least modified in the confirmation process. In another example, a rating function or method can assign a higher rating to anticipatory Instruction Sets 526 that were canceled least number of times by User 50. Any other automatic rating function or method can be utilized. In yet other embodiments, Confirmation Unit 550 can serve as a means of canceling anticipatory Instruction Sets 526 if

they did not match User's 50 intended operation. For example, Decision-making Unit 540 may determine one or more anticipatory Instruction Sets 526, which the system may automatically execute. The system may save the state of Computing Device 70, Processor 11 (save its register values, etc.), Application Program 18 (i.e. save its variables, data structures, objects, location of its current instruction, etc.), Avatar 605, and/or other processing elements before executing anticipatory Instruction Sets 526. User 50 may be provided with an interface (i.e. graphical user interface, selectable list of prior executed anticipatory Instruction Sets 526, etc.) to cancel one or more of the prior executed anticipatory Instruction Sets 526, and restore Computing Device 70, Processor 11, Application Program 18, Avatar 605, and/or other processing elements to a prior state. In some aspects, Confirmation Unit 550 can optionally be disabled or omitted in order to provide an uninterrupted operation of Avatar 605, Processor 11, and/or Application Program 18. For example, a form based application may be suitable for implementing the user confirmation step, whereas, a game application may be less suitable for implementing such interrupting step due to the real time nature of game application's operation.

Referring to Fig. 12, an embodiment of Knowledge Structuring Unit 520 correlating individual Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527 is illustrated. Knowledge Structuring Unit 520 comprises the functionality for structuring the knowledge of Avatar's 605 operation in circumstances including objects with various properties, and/or other functionalities. Knowledge Structuring Unit 520 comprises the functionality for correlating one or more Collections of Object Representations 525 representing Objects 615 in Avatar's 605 surrounding with any Instruction Sets 526 and/or Extra Info 527. The Instruction Sets 526 may be used or executed in operating Avatar 605. Knowledge Structuring Unit 520 comprises the functionality for creating or generating Knowledge Cell 800 and storing one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 into the Knowledge Cell 800. As such, Knowledge Cell 800 comprises the functionality for storing one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527. Knowledge Cell 800 includes knowledge (i.e. unit of knowledge, etc.) of how Avatar 605 operated in a circumstance including objects with various properties. Once created or generated, Knowledge Cells 800 can be used in/as neurons, nodes, vertices, or other elements in any of the data structures or arrangements (i.e. neural networks, graphs, sequences, etc.) used for storing the knowledge of Avatar's 605 operation in circumstances including objects with various properties, thereby facilitating learning functionalities herein. It should be noted that Extra Info 527 may be optionally used in some implementations to enable enhanced comparisons or decision making in autonomous Avatar 605 operation where applicable, and that Extra Info 527 can be omitted in alternate implementations.

In some embodiments, Knowledge Structuring Unit 520 receives one or more Collections of Object Representations 525 from Object Processing Unit 140. Knowledge Structuring Unit 520 may also receive one or more Instruction Sets 526 from Acquisition Interface 120. Knowledge Structuring Unit 520 may further receive any Extra Info 527. Although, Extra Info 527 is not shown in this and/or other figures for clarity of illustration, it should be noted that any Collection of Object Representations 525, Instruction Set 526, and/or other element may include or be associated with Extra Info 527. Knowledge Structuring Unit 520 may correlate one or more Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527. Knowledge Structuring Unit 520 may then create Knowledge Cell 800 and store the one or more Collections of Object Representations 525 correlated with Instruction Sets 526 and/or Extra Info 527 into the Knowledge Cell 800. Knowledge Cell 800 may include any data

structure or arrangement that can facilitate such storing. For example, Knowledge Structuring Unit 520 may create Knowledge Cell 800ax and structure within it Collection of Object Representations 525a1 correlated with Instruction Sets 526a1-526a3 and/or any Extra Info 527 (not shown). Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax a Collection of Object Representations 525a2 correlated with Instruction Set 526a4 and/or any Extra Info 527 (not shown). Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax a Collection of Object Representations 525a3 without a correlated Instruction Set 526 and/or Extra Info 527. Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax a Collection of Object Representations 525a4 correlated with Instruction Sets 526a5-526a6 and/or any Extra Info 527 (not shown). Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax a Collection of Object Representations 525a5 without a correlated Instruction Set 526 and/or Extra Info 527. Knowledge Structuring Unit 520 may structure within Knowledge Cell 800ax additional Collections of Object Representations 525 correlated with any number (including zero [i.e. uncorrelated]) of Instruction Sets 526 and/or Extra Info 527 by following similar logic as described above.

In some embodiments, Knowledge Structuring Unit 520 may correlate a Collection of Object Representations 525 with one or more temporally corresponding Instruction Sets 526 and/or Extra Info 527. This way, Knowledge Structuring Unit 520 can structure the knowledge of Avatar's 605 operation at or around the time of generating Collections of Object Representations 525. Such functionality enables spontaneous or seamless learning of Avatar's 605 operation in circumstances including objects with various properties as Avatar 605 is operated in real time. In some designs, Knowledge Structuring Unit 520 may receive a stream of Instruction Sets 526 used or executed to effect Avatar's 605 operations as well as a stream of Collections of Object Representations 525 representing Objects 615 in Avatar's 605 surrounding as the operations are performed. Knowledge Structuring Unit 520 can then correlate Collections of Object Representations 525 from the stream of Collections of Object Representations 525 with temporally corresponding Instruction Sets 526 from the stream of Instruction Sets 526 and/or any Extra Info 527. Collections of Object Representations 525 without a temporally corresponding Instruction Set 526 may be uncorrelated, for instance. In some aspects, Instruction Sets 526 and/or Extra Info 527 that temporally correspond to a Collection of Object Representations 525 may include Instruction Sets 526 used and/or Extra Info 527 obtained at the time of generating the Collection of Object Representations 525. In other aspects, Instruction Sets 526 and/or Extra Info 527 that temporally correspond to a Collection of Object Representations 525 may include Instruction Sets 526 used and/or Extra Info 527 obtained within a certain time period before and/or after generating the Collection of Object Representations 525. For example, Instruction Sets 526 and/or Extra Info 527 that temporally correspond to a Collection of Object Representations 525 may include Instruction Sets 526 used and/or Extra Info 527 obtained within 50 milliseconds, 1 second, 3 seconds, 20 seconds, 1 minute, 41 minutes, 2 hours, or any other time period before and/or after generating the Collection of Object Representations 525. Such time periods can be defined by a user, by ACAAO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. In other aspects, Instruction Sets 526 and/or Extra Info 527 that temporally correspond to a Collection of Object Representations 525 may include Instruction Sets 526 used and/or Extra Info 527 obtained from the time of generating the Collection of Object Representations 525 to the time of generating a next Collection of Object Representations 525. In further aspects, Instruction Sets 526 and/or Extra Info 527 that temporally correspond to a Collection of Object

Representations 525 may include Instruction Sets 526 used and/or Extra Info 527 obtained from the time of generating a previous Collection of Object Representations 525 to the time of generating the Collection of Object Representations 525. Any other temporal relationship or correspondence between Collections of Object Representations 525 and correlated Instruction Sets 526 and/or Extra Info 527 can be implemented.

In some embodiments, Knowledge Structuring Unit 520 can structure the knowledge of Avatar's 605 operation in a circumstance including objects with various properties into any number of Knowledge Cells 800. In some aspects, Knowledge Structuring Unit 520 can structure into a Knowledge Cell 800 a single Collection of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527. In other aspects, Knowledge Structuring Unit 520 can structure into a Knowledge Cell 800 any number (i.e. 2, 3, 6, 9, 21, 98, 3210, 13592, 513299, 9147224, etc.) of Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527. In a special case, Knowledge Structuring Unit 520 can structure all Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 into a single long Knowledge Cell 800. In further aspects, Knowledge Structuring Unit 520 can structure Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 into a plurality of Knowledge Cells 800. In a special case, Knowledge Structuring Unit 520 can store periodic streams of Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 into a plurality of Knowledge Cells 800 such as hourly, daily, weekly, monthly, yearly, or other periodic Knowledge Cells 800.

Referring to Fig. 13, another embodiment of Knowledge Structuring Unit 520 correlating individual Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527 is illustrated. In such embodiments, Knowledge Structuring Unit 520 may generate Knowledge Cells 800 each comprising a single Collection of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527.

Referring to Fig. 14, an embodiment of Knowledge Structuring Unit 520 correlating streams of Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527 is illustrated. For example, Knowledge Structuring Unit 520 may create Knowledge Cell 800ax and structure within it a stream of Collections of Object Representations 525a1-525an correlated with Instruction Set 526a1 and/or any Extra Info 527 (not shown). Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax a stream of Collections of Object Representations 525b1-525bn correlated with Instruction Sets 526a2-526a4 and/or any Extra Info 527 (not shown). Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax a stream of Collections of Object Representations 525c1-525cn without correlated Instruction Sets 526 and/or Extra Info 527. Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax a stream of Collections of Object Representations 525d1-525dn correlated with Instruction Sets 526a5-526a6 and/or any Extra Info 527 (not shown). Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax additional streams of Collections of Object Representations 525 correlated with any number (including zero [i.e. uncorrelated]) of Instruction Sets 526 and/or Extra Info 527 by following similar logic as described above. The number of Collections of Object Representations 525 in some or all streams of Collections of Object Representations 525a1-525an, 525b1-525bn, 525c1-525cn, 525d1-525dn, etc. may be equal or different. It should be noted that n or other such letters or indicia may follow the sequence and/or context where they are indicated. Also, a same letter or indicia such as n may represent a different number in different sequences or elements of a drawing.

Referring to Fig. 15, another embodiment of Knowledge Structuring Unit 520 correlating streams of Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527 is illustrated. In such embodiments, Knowledge Structuring Unit 520 may generate Knowledge Cells 800 each comprising a single stream of Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527.

Knowledgebase 530 comprises the functionality for storing knowledge of Avatar's 605 operation in circumstances including objects with various properties, and/or other functionalities. Knowledgebase 530 comprises the functionality for storing one or more Collections of Object Representations 525 representing Objects 615 in Avatar's 605 surrounding correlated with any Instruction Sets 526 and/or Extra Info 527. The Instruction Sets 526 may be used or executed in operating Avatar 605. Knowledgebase 530 comprises the functionality for storing one or more Knowledge Cells 800 each including one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527. In some aspects, Collections of Object Representations 525 correlated with Instruction Sets 526 and/or Extra Info 527 can be stored directly within Knowledgebase 530 without using Knowledge Cells 800 as the intermediary data structures. In some embodiments, Knowledgebase 530 may be or include Neural Network 530a (later described). In other embodiments, Knowledgebase 530 may be or include Graph 530b (later described). In further embodiments, Knowledgebase 530 may be or include Collection of Sequences 530c (later described). In further embodiments, Knowledgebase 530 may be or include Sequence 533 (later described). In further embodiments, Knowledgebase 530 may be or include Collection of Knowledge Cells 530d (later described). In general, Knowledgebase 530 may be or include any data structure or arrangement capable of storing knowledge of Avatar's 605 operation in circumstances including objects with various properties. Knowledgebase 530 may reside locally on Computing Device 70, or remotely (i.e. remote Knowledgebase 530, etc.) on a remote computing device (i.e. server, cloud, etc.) accessible over a network or an interface.

In some embodiments, Knowledgebase 530 from one Computing Device 70 or ACAAO Unit 100 can be transferred to one or more other Computing Devices 70 or ACAAO Units 100. Therefore, the knowledge of Avatar's 605 operation in circumstances including objects with various properties learned and/or stored on one Computing Device 70 or ACAAO Unit 100 can be transferred to one or more other Computing Devices 70 or ACAAO Units 100. In one example, Knowledgebase 530 can be copied or downloaded to a file or other repository from one Computing Device 70 or ACAAO Unit 100 and loaded or inserted into another Computing Device 70 or ACAAO Unit 100. In another example, Knowledgebase 530 from one Computing Device 70 or ACAAO Unit 100 can be available on a server accessible by other Computing Devices 70 or ACAAO Units 100 over a network or an interface. Once loaded into or accessed by a receiving Computing Device 70 or ACAAO Unit 100, the receiving Computing Device 70 or ACAAO Unit 100 can then implement the knowledge of Avatar's 605 operation in circumstances including objects with various properties learned or stored on the originating Computing Device 70 or ACAAO Unit 100.

In some embodiments, multiple Knowledgebases 530 (i.e. Knowledgebases 530 from different Computing Devices 70 or ACAAO Units 100, etc.) can be combined to accumulate collective knowledge of operating Avatar 605 in circumstances including objects with various properties. In one example, one Knowledgebase 530 can be appended to another Knowledgebase 530 such as appending one Collection of Sequences 530c (later described) to another Collection of Sequences 530c, appending one Sequence 533 (later described) to another Sequence 533, appending one Collection of Knowledge Cells 530d (later described) to another Collection of Knowledge Cells 530d, and/or appending other data structures or elements thereof. In another example, one Knowledgebase 530 can be

copied into another Knowledgebase 530 such as copying one Collection of Sequences 530c into another Collection of Sequences 530c, copying one Collection of Knowledge Cells 530d into another Collection of Knowledge Cells 530d, and/or copying other data structures or elements thereof. In a further example, in the case of Knowledgebase 530 being or including Graph 530b or graph-like data structure (i.e. Neural Network 530a, tree, etc.), a union can be utilized to combine two or more Graphs 530b or graph-like data structures. For instance, a union of two Graphs 530b or graph-like data structures may include a union of their vertex (i.e. node, etc.) sets and their edge (i.e. connection, etc.) sets. Any other operations or combination thereof on graphs or graph-like data structures can be utilized to combine Graphs 530b or graph-like data structures. In a further example, one Knowledgebase 530 can be combined with another Knowledgebase 530 through later described learning processes where Knowledge Cells 800 may be applied one at a time and connected with prior and/or subsequent Knowledge Cells 800 such as in Graph 530b or Neural Network 530a. In such embodiments, instead of Knowledge Cells 800 generated by Knowledge Structuring Unit 520, the learning process may utilize Knowledge Cells 800 from one Knowledgebase 530 to apply them onto another Knowledgebase 530. Any other techniques known in art including custom techniques for combining data structures can be utilized for combining Knowledgebases 530 in alternate implementations. In any of the aforementioned and/or other combining techniques, similarity of elements (i.e. nodes/vertices, edges/connections, etc.) can be utilized in determining whether an element from one Knowledgebase 530 matches an element from another Knowledgebase 530, and substantially or otherwise similar elements may be considered a match for combining purposes in some designs. Any features, functionalities, and embodiments of Similarity Comparison 125 (later described) can be used in such similarity determinations. A combined Knowledgebase 530 can be offered as a network service (i.e. online application, etc.), downloadable file, or other repository to all ACAAO Units 100 configured to utilize the combined Knowledgebase 530. For example, a computer game (i.e. Application Program 18, etc.) including or interfaced with ACAAO Unit 100 having access to a combined Knowledgebase 530 can use a collective knowledge for Avatar's 605 operation in circumstances including objects with various properties learned from multiple players (i.e. Users 50, etc.) for Avatar's 605 autonomous operation.

Referring to Fig. 16, the disclosed artificially intelligent devices, systems, and methods for learning and/or using an avatar's circumstances for autonomous avatar operation may include various artificial intelligence models and/or techniques. The disclosed devices, systems, and methods are independent of the artificial intelligence model and/or technique used and any model and/or technique can be utilized to facilitate the functionalities described herein. Examples of these models and/or techniques include deep learning, supervised learning, unsupervised learning, neural networks (i.e. convolutional neural network, recurrent neural network, deep neural network, etc.), search-based, logic and/or fuzzy logic-based, optimization-based, tree/graph/other data structure-based, hierarchical, symbolic and/or sub-symbolic, evolutionary, genetic, multi-agent, deterministic, probabilistic, statistical, and/or other models and/or techniques.

In one example shown in Model A, the disclosed artificially intelligent devices, systems, and methods for learning and/or using an avatar's circumstances for autonomous avatar operation may include a neural network (also referred to as artificial neural network, etc.). As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include a network of Nodes 852 (also referred to as neurons, etc.) and Connections 853 similar to that of a brain. Node 852

can store any data, object, data structure, and/or other item, or reference thereto. Node 852 may also include a function for transforming or manipulating any data, object, data structure, and/or other item. Examples of such transformation functions include mathematical functions (i.e. addition, subtraction, multiplication, division, sin, cos, log, derivative, integral, etc.), object manipulation functions (i.e. creating an object, modifying an object, deleting an object, appending objects, etc.), data structure manipulation functions (i.e. creating a data structure, modifying a data structure, deleting a data structure, creating a data field, modifying a data field, deleting a data field, etc.), and/or other transformation functions. Connection 853 may include or be associated with a value such as a symbolic label or numeric attribute (i.e. weight, cost, capacity, length, etc.). A computational model can be utilized to compute values from inputs based on a pre-programmed or learned function or method. For example, a neural network may include one or more input neurons that can be activated by inputs. Activations of these neurons can then be passed on, weighted, and transformed by a function to other neurons. Neural networks may range from those with only one layer of single direction logic to multi-layer of multi-directional feedback loops. A neural network can use weights to change the parameters of the network's throughput. A neural network can learn by input from its environment or from self-teaching using written-in rules. A neural network can be utilized as a predictive modeling approach in machine learning. An exemplary embodiment of a neural network (i.e. Neural Network 530a, etc.) is described later.

In another example shown in Model B, the disclosed artificially intelligent devices, systems, and methods for learning and/or using an avatar's circumstances for autonomous avatar operation may include a graph or graph-like data structure. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include Nodes 852 (also referred to as vertices or points, etc.) and Connections 853 (also referred to as edges, arrows, lines, arcs, etc.) organized as a graph. In general, any Node 852 in a graph can be connected to any other Node 852. A Connection 853 may include unordered pair of Nodes 852 in an undirected graph or ordered pair of Nodes 852 in a directed graph. Nodes 852 can be part of the graph structure or external entities represented by indices or references. A graph can be utilized as a predictive modeling approach in machine learning. Nodes 852, Connections 853, and/or other elements or operations of a graph may include any features, functionalities, and embodiments of the aforementioned Nodes 852, Connections 853, and/or other elements or operations of a neural network, and vice versa. An exemplary embodiment of a graph (i.e. Graph 530b, etc.) is described later.

In a further example shown in Model C, the disclosed artificially intelligent devices, systems, and methods for learning and/or using an avatar's circumstances for autonomous avatar operation may include a tree or tree-like data structure. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include Nodes 852 and Connections 853 (also referred to as references, edges, etc.) organized as a tree. In general, a Node 852 in a tree can be connected to any number (i.e. including zero, etc.) of children Nodes 852. A tree can be utilized as a predictive modeling approach in machine learning. Nodes 852, Connections 853, and/or other elements or operations of a tree may include any features, functionalities, and embodiments of the aforementioned Nodes 852, Connections 853, and/or other elements or operations of a neural network and/or graph, and vice versa.

In a further example shown in Model D, the disclosed artificially intelligent devices, systems, and methods for learning and/or using an avatar's circumstances for autonomous avatar operation may include a sequence or sequence-like data structure. As such, machine learning, knowledge structuring or representation, decision making,

pattern recognition, and/or other artificial intelligence functionalities may include a structure of Nodes 852 and/or Connections 853 organized as a sequence. In some aspects, Connections 853 may be optionally omitted from a sequence as the sequential order of Nodes 852 in a sequence may be implied in the structure. A sequence can be utilized as a predictive modeling approach in machine learning. Nodes 852, Connections 853, and/or other elements or operations of a sequence may include any features, functionalities, and embodiments of the aforementioned Nodes 852, Connections 853, and/or other elements or operations of a neural network, graph, and/or tree, and vice versa. An exemplary embodiment of a sequence (i.e. Collection of Sequences 530c, Sequence 533, etc.) is described later.

In yet another example, the disclosed artificially intelligent devices, systems, and methods for learning and/or using an avatar's circumstances for autonomous avatar operation may include a search-based model and/or technique. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include searching through a collection of possible solutions. For example, a search method can search through a neural network, graph, tree, sequence, or other data structure that includes data elements of interest. A search may use heuristics to limit the search for solutions by eliminating choices that are unlikely to lead to the goal. Heuristic techniques may provide a best guess solution. A search can also include optimization. For example, a search may begin with a guess and then refine the guess incrementally until no more refinements can be made. In a further example, the disclosed systems, devices, and methods may include logic-based model and/or technique. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities can use formal or other type of logic. Logic based models may involve making inferences or deriving conclusions from a set of premises. As such, a logic based system can extend existing knowledge or create new knowledge automatically using inferences. Examples of the types of logic that can be utilized include propositional or sentential logic that comprises logic of statements which can be true or false; first-order logic that allows the use of quantifiers and predicates and that can express facts about objects, their properties, and their relations with each other; fuzzy logic that allows degrees of truth to be represented as a value between 0 and 1 rather than simply 0 (false) or 1 (true), which can be used for uncertain reasoning; subjective logic that comprises a type of probabilistic logic that may take uncertainty and belief into account, which can be suitable for modeling and analyzing situations involving uncertainty, incomplete knowledge, and different world views; and/or other types of logic. In a further example, the disclosed systems, devices, and methods may include a probabilistic model and/or technique. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities can be implemented to operate with incomplete or uncertain information where probabilities may affect outcomes. Bayesian network, among other models, is an example of a probabilistic tool used for purposes such as reasoning, learning, planning, perception, and/or others. One of ordinary skill in art will understand that the aforementioned artificial intelligence models and/or techniques are described merely as examples of a variety of possible implementations, and that while all possible artificial intelligence models and/or techniques are too voluminous to describe, other artificial intelligence models and/or techniques known in art are within the scope of this disclosure. One of ordinary skill in art will also recognize that an intelligent system may solve a specific problem by using any model and/or technique that works such as, for example, some systems can be symbolic and logical, some can be sub-symbolic neural networks, some can be deterministic or probabilistic, some

can be hierarchical, some may include searching techniques, some may include optimization techniques, while others may use other or a combination of models and/or techniques. In general, any artificial intelligence model and/or technique can be utilized that can facilitate the functionalities described herein.

Referring to Figs. 17A-17C, embodiments of interconnected Knowledge Cells 800 and updating weights of Connections 853 are illustrated. As shown for example in Fig. 17A, Knowledge Cell 800za is connected to Knowledge Cell 800zb and Knowledge Cell 800zc by Connection 853z1 and Connection 853z2, respectively. Each of Connection 853z1 and Connection 853z2 may include or be associated with occurrence count, weight, and/or other parameters or data. The number of occurrences may track or store the number of observations that a Knowledge Cell 800 was followed by another Knowledge Cell 800 indicating a connection or relationship between them. For example, Knowledge Cell 800za was followed by Knowledge Cell 800zb 10 times as indicated by the number of occurrences of Connection 853z1. Also, Knowledge Cell 800za was followed by Knowledge Cell 800zc 15 times as indicated by the number of occurrences of Connection 853z2. The weight of Connection 853z1 can be calculated or determined as the number of occurrences of Connection 853z1 divided by the sum of occurrences of all connections (i.e. Connection 853z1 and Connection 853z2, etc.) originating from Knowledge Cell 800za. Therefore, the weight of Connection 853z1 can be calculated or determined as 10/(10+15)=0.4, for example. Also, the weight of Connection 853z2 can be calculated or determined as 15/(10+15)=0.6, for example. Therefore, the sum of weights of Connection 853z1, Connection 853z2, and/or any other Connections 853 originating from Knowledge Cell 800za may equal to 1 or 100%. As shown for example in Fig. 17B, in the case that Knowledge Cell 800zd is inserted and an observation is made that Knowledge Cell 800zd follows Knowledge Cell 800za, Connection 853z3 can be created between Knowledge Cell 800za and Knowledge Cell 800zd. The occurrence count of Connection 853z3 can be set to 1 and weight determined as 1/(10+15+1)=0.038. The weights of all other connections (i.e. Connection 853z1, Connection 853z2, etc.) originating from Knowledge Cell 800za may be updated to account for the creation of Connection 853z3. Therefore, the weight of Connection 853z1 can be updated as 10/(10+15+1)=0.385. The weight of Connection 853z2 can also be updated as 15/(10+15+1)=0.577. As shown for example in Fig. 17C, in the case that an additional occurrence of Connection 853z1 is observed (i.e. Knowledge Cell 800zb followed Knowledge Cell 800za, etc.), occurrence count of Connection 853z1 and weights of all connections (i.e. Connection 853z1, Connection 853z2, and Connection 853z3, etc.) originating from Knowledge Cell 800za may be updated to account for this observation. The occurrence count of Connection 853z1 can be increased by 1 and its weight updated as 11/(11+15+1)=0.407. The weight of Connection 853z2 can also be updated as 15/(11+15+1)=0.556. The weight of Connection 853z3 can also be updated as 1/(11+15+1)=0.037.

Referring to Fig. 18, an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Collection of Knowledge Cells 530d is illustrated. Collection of Knowledge Cells 530d comprises the functionality for storing any number of Knowledge Cells 800. In some aspects, Knowledge Cells 800 may be stored into or applied onto Collection of Knowledge Cells 530d in a learning or training process. In effect, Collection of Knowledge Cells 530d may store Knowledge Cells 800 that can later be used to enable autonomous Avatar 605 operation. In some embodiments, Knowledge Structuring Unit 520 structures or generates Knowledge Cells 800 as previously described and the system applies them onto Collection of Knowledge Cells 530d, thereby implementing learning Avatar's 605 operation in circumstances including objects with various properties. The term apply or applying may

refer to storing, copying, inserting, updating, or other similar action, therefore, these terms may be used interchangeably herein depending on context. The system can perform Similarity Comparisons 125 (later described) of a newly structured Knowledge Cell 800 from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Collection of Knowledge Cells 530d. If a substantially similar Knowledge Cell 800 is not found in Collection of Knowledge Cells 530d, the system may insert (i.e. copy, store, etc.) the Knowledge Cell 800 from Knowledge Structuring Unit 520 into Collection of Knowledge Cells 530d, for example. On the other hand, if a substantially similar Knowledge Cell 800 is found in Collection of Knowledge Cells 530d, the system may optionally omit inserting the Knowledge Cell 800 from Knowledge Structuring Unit 520 as inserting a substantially similar Knowledge Cell 800 may not add much or any additional knowledge to the Collection of Knowledge Cells 530d, for example. Also, inserting a substantially similar Knowledge Cell 800 can optionally be omitted to save storage resources and limit the number of Knowledge Cells 800 that may later need to be processed or compared. Any features, functionalities, and embodiments of Similarity Comparison 125, importance index (later described), similarity index (later described), and/or other disclosed elements can be utilized to facilitate determination of substantial or other similarity and whether to insert a newly structured Knowledge Cell 800 into Collection of Knowledge Cells 530d.

For example, the system can perform Similarity Comparisons 125 (later described) of Knowledge Cell 800ba from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Collection of Knowledge Cells 530d. In the case that a substantially similar match is found between Knowledge Cell 800ba and any of the Knowledge Cells 800 in Collection of Knowledge Cells 530d, the system may perform no action. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bb from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Collection of Knowledge Cells 530d. In the case that a substantially similar match is not found, the system may insert a new Knowledge Cell 800 into Collection of Knowledge Cells 530d and copy Knowledge Cell 800bb into the inserted new Knowledge Cell 800. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bc from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Collection of Knowledge Cells 530d. In the case that a substantially similar match is found between Knowledge Cell 800bc and any of the Knowledge Cells 800 in Collection of Knowledge Cells 530d, the system may perform no action. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bd from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Collection of Knowledge Cells 530d. In the case that a substantially similar match is not found, the system may insert a new Knowledge Cell 800 into Collection of Knowledge Cells 530d and copy Knowledge Cell 800bd into the inserted new Knowledge Cell 800. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800be from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Collection of Knowledge Cells 530d. In the case that a substantially similar match is not found, the system may insert a new Knowledge Cell 800 into Collection of Knowledge Cells 530d and copy Knowledge Cell 800be into the inserted new Knowledge Cell 800. Applying any additional Knowledge Cells 800 from Knowledge Structuring Unit 520 onto Collection of Knowledge Cells 530d follows similar logic or process as the above-described.

Referring to Fig. 19, an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Neural Network 530a is illustrated. Neural Network 530a includes a number of neurons or Nodes 852 interconnected by Connections 853 as previously described. Knowledge Cells 800 are shown instead of Nodes 852 to simplify the illustration as Node 852 includes a Knowledge Cell 800, for example. Therefore, Knowledge Cells 800 and Nodes 852 can be

used interchangeably herein depending on context. It should be noted that Node 852 may include other elements and/or functionalities instead of or in addition to Knowledge Cell 800. In some aspects, Knowledge Cells 800 may be stored into or applied onto Neural Network 530a individually or collectively in a learning or training process. In some designs, Neural Network 530a comprises a number of Layers 854 each of which may include one or more Knowledge Cells 800. Knowledge Cells 800 in successive Layers 854 can be connected by Connections 853. Connection 853 may include or be associated with occurrence count, weight, and/or other parameter or data as previously described. Neural Network 530a may include any number of Layers 854 comprising any number of Knowledge Cells 800. In some aspects, Neural Network 530a may store Knowledge Cells 800 interconnected by Connections 853 where following a path through the Neural Network 530a can later be used to enable autonomous Avatar 605 operation. It should be understood that, in some embodiments, Knowledge Cells 800 in one Layer 854 of Neural Network 530a need not be connected only with Knowledge Cells 800 in a successive Layer 854, but also in any other Layer 854, thereby creating shortcuts (i.e. shortcut Connections 853, etc.) through Neural Network 530a. A Knowledge Cell 800 can also be connected to itself such as, for example, in recurrent neural networks. In general, any Knowledge Cell 800 can be connected with any other Knowledge Cell 800 anywhere else in Neural Network 530a. In further embodiments, back-propagation of any data or information can be implemented. In one example, back-propagation of similarity (i.e. similarity index, etc.) of compared Knowledge Cells 800 in a path through Neural Network 530a can be implemented. In another example, back-propagation of errors can be implemented. Such back-propagations can then be used to adjust occurrence counts and/or weights of Connections 853 for better future predictions, for example. Any other back-propagation can be implemented for other purposes. Any combination of Nodes 852 (i.e. Nodes 852 comprising Knowledge Cells 800, etc.), Connections 853, Layers 854, and/or other elements or techniques can be implemented in alternate embodiments. Neural Network 530a may include any type or form of a neural network known in art such as a feed-forward neural network, a back-propagating neural network, a recurrent neural network, a convolutional neural network, deep neural network, and/or others including a custom neural network.

In some embodiments, Knowledge Structuring Unit 520 structures or generates Knowledge Cells 800 and the system applies them onto Neural Network 530a, thereby implementing learning Avatar's 605 operation in circumstances including objects with various properties. The system can perform Similarity Comparisons 125 (later described) of a Knowledge Cell 800 from Knowledge Structuring Unit 520 with Knowledge Cells 800 in a Layer 854 of Neural Network 530a. If a substantially similar Knowledge Cell 800 is not found in the Layer 854 of Neural Network 530a, the system may insert (i.e. copy, store, etc.) the Knowledge Cell 800 from Knowledge Structuring Unit 520 into the Layer 854 of Neural Network 530a, and create a Connection 853 to the inserted Knowledge Cell 800 from a Knowledge Cell 800 in a prior Layer 854 including assigning an occurrence count to the new Connection 853, calculating a weight of the new Connection 853, and updating any other Connections 853 originating from the Knowledge Cell 800 in the prior Layer 854. On the other hand, if a substantially similar Knowledge Cell 800 is found in the Layer 854 of Neural Network 530a, the system may update occurrence count and weight of Connection 853 to that Knowledge Cell 800 from a Knowledge Cell 800 in a prior Layer 854, and update any other Connections 853 originating from the Knowledge Cell 800 in the prior Layer 854.

For example, the system can perform Similarity Comparisons 125 (later described) of Knowledge Cell 800ba from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Layer 854a of Neural Network 530a. In the

case that a substantially similar match is found between Knowledge Cell 800ba and Knowledge Cell 800ea, the system may perform no action since Knowledge Cell 800ea is the initial Knowledge Cell 800. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bb from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Layer 854b of Neural Network 530a. In the case that a substantially similar match is found between Knowledge Cell 800bb and Knowledge Cell 800eb, the system may update occurrence count and weight of Connection 853e1 between Knowledge Cell 800ea and Knowledge Cell 800eb, and update weights of other Connections 853 originating from Knowledge Cell 800ea as previously described. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bc from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Layer 854c of Neural Network 530a. In the case that a substantially similar match is not found, the system may insert Knowledge Cell 800ec into Layer 854c and copy Knowledge Cell 800bc into the inserted Knowledge Cell 800ec. The system may also create Connection 853e2 between Knowledge Cell 800eb and Knowledge Cell 800ec with occurrence count of 1 and weight calculated based on the occurrence count as previously described. The system may also update weights of other Connections 853 (one in this example) originating from Knowledge Cell 800eb as previously described. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bd from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Layer 854d of Neural Network 530a. In the case that a substantially similar match is not found, the system may insert Knowledge Cell 800ed into Layer 854d and copy Knowledge Cell 800bd into the inserted Knowledge Cell 800ed. The system may also create Connection 853e3 between Knowledge Cell 800ec and Knowledge Cell 800ed with occurrence count of 1 and weight of 1. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800be from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Layer 854e of Neural Network 530a. In the case that a substantially similar match is not found, the system may insert Knowledge Cell 800ee into Layer 854e and copy Knowledge Cell 800be into the inserted Knowledge Cell 800ee. The system may also create Connection 853e4 between Knowledge Cell 800ed and Knowledge Cell 800ee with occurrence count of 1 and weight of 1. Applying any additional Knowledge Cells 800 from Knowledge Structuring Unit 520 onto Neural Network 530a follows similar logic or process as the above-described.

Referring now to Similarity Comparison 125, Similarity Comparison 125 comprises the functionality for comparing or matching Knowledge Cells 800 or portions thereof, and/or other functionalities. Similarity Comparison 125 comprises the functionality for comparing or matching Collections of Object Representations 525 or portions thereof. Similarity Comparison 125 comprises the functionality for comparing or matching streams of Collections of Object Representations 525 or portions thereof. Similarity Comparison 125 comprises the functionality for comparing or matching Object Representations 625 or portions thereof. Similarity Comparison 125 comprises the functionality for comparing or matching Object Properties 630 or portions thereof. Similarity Comparison 125 comprises the functionality for comparing or matching Instruction Sets 526, Extra Info 527, text (i.e. characters, words, phrases, etc.), numbers, and/or other elements or portions thereof. Similarity Comparison 125 may include functions, rules, and/or logic for performing matching or comparisons and for determining that while a perfect match is not found, a partial or similar match has been found. In some aspects, a partial match may include a substantially or otherwise similar match, and vice versa. Therefore, these terms may be used interchangeably herein depending on context. As such, Similarity Comparison 125 may include determining substantial similarity or substantial match of compared elements. Although, substantial similarity or substantial match is frequently used herein, it should be understood that

any level of similarity, however high or low, may be utilized as defined by the rules (i.e. thresholds, etc.) for similarity. The rules for similarity or similar match can be defined by a user, by ACAAO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. In some designs, Similarity Comparison 125 comprises the functionality to automatically define appropriately strict rules for determining similarity of the compared elements. Similarity Comparison 125 can therefore set, reset, and/or adjust the strictness of the rules for finding or determining similarity of the compared elements, thereby fine tuning Similarity Comparison 125 so that the rules for determining similarity are appropriately strict. In some aspects, the rules for determining similarity may include a similarity threshold. As such, Similarity Comparison 125 can determine similarity of compared elements if their similarity exceeds a similarity threshold. In other aspects, the rules for determining similarity may include a difference threshold. As such, Similarity Comparison 125 can determine similarity of compared elements if their difference is lower than a difference threshold. In further aspects, the rules for determining similarity may include other thresholds. Similarity Comparison 125 enables comparing circumstances including objects with various properties and determining their similarity or match. In one example, a circumstance including an object located at a distance of 9m and an angle/bearing of 97° relative to Avatar 605 may be found similar or matching by Similarity Comparison 125 to a circumstance including the same or similar object located at a distance of 8.7m and an angle/bearing of 101° relative to Avatar 605. In another example, a circumstance including an object detected as a female person may be found similar or matching by Similarity Comparison 125 to a circumstance including an object detected as a male person. In general, any one or more properties (i.e. existence, type, identity, distance, bearing/angle, location, shape/size, activity, etc.) of one or more objects can be utilized for determining similarity or match of circumstances including objects with various properties. Therefore, Similarity Comparison 125 provides flexibility in comparing and determining similarity of a variety of possible circumstances of Avatar 605.

In some embodiments where compared Knowledge Cells 800 include a single Collection of Object Representations 525, in determining similarity of Knowledge Cells 800, Similarity Comparison 125 can perform comparison of individual Collections of Object Representations 525 or portions (i.e. Object Representations 625, Object Properties 630, etc.) thereof such as comparison of Collection of Object Representations 525 or portions thereof from one Knowledge Cell 800 with Collection of Object Representations 525 or portions thereof from another Knowledge Cell 800. In some aspects, total equivalence is achieved when Collection of Object Representations 525 or portions thereof from one Knowledge Cell 800 matches Collection of Object Representations 525 or portions thereof from another Knowledge Cell 800. If total equivalence is not found, Similarity Comparison 125 may attempt to determine substantial or other similarity of compared Knowledge Cells 800.

In some embodiments, in determining substantial similarity of individually compared Collections of Object Representations 525 (i.e. Collections of Object Representations 525 from the compared Knowledge Cells 800, etc.), Similarity Comparison 125 can compare one or more Object Representations 625 or portions (i.e. Object Properties 630, etc.) thereof from one Collection of Object Representations 525 with one or more Object Representations 625 or portions thereof from another Collection of Object Representations 525. In some aspects, total equivalence is found when all Object Representations 625 or portions thereof from one Collection of Object Representations 525 match all Object Representations 625 or portions thereof from another Collection of Object Representations 525. In other aspects, if total equivalence is not found, Similarity Comparison 125 may attempt to determine substantial

similarity of compared Collections of Object Representations 525. In one example, substantial similarity can be achieved when most of the Object Representations 625 or portions thereof from the compared Collections of Object Representations 525 match or substantially match. In another example, substantial similarity can be achieved when at least a threshold number (i.e. 1, 2, 4, 7, 18, etc.) or percentage (i.e. 41%, 62%, 79%, 85%, 93%, etc.) of Object Representations 625 or portions thereof from the compared Collections of Object Representations 525 match or substantially match. Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching Object Representations 625 or portions thereof from the compared Collections of Object Representations 525 exceeds a threshold number (i.e. 1, 2, 4, 7, 18, etc.) or a threshold percentage (i.e. 41%, 62%, 79%, 85%, 93%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or percentage of Object Representations 625 or portions thereof from the compared Collections of Object Representations 525 match or substantially match. Such thresholds can be defined by a user, by ACAAO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In some aspects, Similarity Comparison 125 can utilize the importance (i.e. as indicated by importance index [later described], etc.) of Object Representations 625 or portions thereof for determining substantial similarity of Collections of Object Representations 525. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to more important Object Representations 625 or portions thereof such as Object Representations 625 representing near Objects 615, Object Representations 625 representing large Objects 615, etc., thereby tolerating mismatches in less important Object Representations 625 or portions thereof such as Object Representations 625 representing distant Objects 615, Object Representations 625 representing small Objects 615, etc. In general, any Object Representation 625 or portion thereof can be assigned higher or lower importance. In further aspects, Similarity Comparison 125 can omit some of the Object Representations 625 or portions thereof from the comparison in determining substantial similarity of Collections of Object Representations 525. In one example, Object Representations 625 representing distant Objects 615 can be omitted from comparison. In another example, Object Representations 625 representing small Objects 615 can be omitted from comparison. In general, any Object Representation 625 or portion thereof can be omitted from comparison depending on implementation.

Similarity Comparison 125 can automatically adjust (i.e. increase or decrease) the strictness of the rules for determining substantial similarity of Collections of Object Representations 525. In some aspects, such adjustment in strictness can be done by Similarity Comparison 125 in response to determining that total equivalence of compared Collections of Object Representations 525 had not been found. Similarity Comparison 125 can keep adjusting the strictness rules until a substantial similarity is found. All the rules or settings of substantial similarity can be set, reset, or adjusted by Similarity Comparison 125 in response to another strictness level determination. For example, Similarity Comparison 125 may attempt to find a match or substantial match in a certain percentage (i.e. 83%, etc.) of Object Representations 625 or portions thereof from the compared Collections of Object Representations 525. If the comparison does not determine substantial similarity of compared Collections of Object Representations 525, Similarity Comparison 125 may decide to decrease the strictness of the rules. In response, Similarity Comparison 125 may attempt to find fewer matching or substantially matching Object Representations 625 or portions thereof than in the previous attempt using stricter rules. If the comparison still does not determine substantial similarity of compared Collections of Object Representations 525, Similarity Comparison 125 may further decrease the strictness

(i.e. down to a certain minimum strictness or threshold, etc.) by requiring fewer Object Representations 625 or portions thereof to match or substantially match, thereby further increasing a chance of finding substantial similarity in compared Collections of Object Representations 525.

Where a reference to Object Representation 625 is used herein it should be understood that a portion of Object Representation 625 (i.e. Object Property 630, etc.) or a plurality of Object Representations 625 can be used instead of or in addition to the Object Representation 625. In one example, instead of or in addition to Object Representation 625, Object Properties 630 and/or other portions that constitute an Object Representation 625 can be compared. In another example, instead of or in addition to Object Representation 625, plurality of Object Representations 625 can be compared. As such, any operations, rules, logic, and/or functions operating on Object Representation 625 may similarly apply to any portion of Object Representation 625 and/or a plurality of Object Representations 625 as applicable. In general, whole Object Representations 625, portions of Object Representations 625, and/or pluralities of Object Representations 625, including any operations thereon, can be combined to arrive at desired results. Some or all of the above-described rules, logic, and/or techniques can be utilized alone or in combination with each other or with other rules, logic, and/or techniques. One of ordinary skill in art will recognize that other techniques known in art for determining similarity of Object Representations 625 and/or other data that would be too voluminous to describe are within the scope of this disclosure.

In some embodiments, in determining substantial similarity of Object Representations 625 (i.e. Object Representations 625 from the compared Collections of Object Representations 525, etc.), Similarity Comparison 125 can compare Object Properties 630 or portions (i.e. characters, words, numbers, etc.) thereof from one Object Representation 625 with Object Properties 630 or portions thereof from another Object Representation 625. In some aspects, total equivalence is found when all Object Properties 630 or portions thereof of one Object Representation 625 match all Object Properties 630 or portions thereof of another Object Representation 625. In other aspects, if total equivalence is not found, Similarity Comparison 125 may attempt to determine substantial similarity of compared Object Representations 625. In one example, substantial similarity can be achieved when most of the Object Properties 630 or portions thereof from the compared Object Representations 625 match or substantially match. In another example, substantial similarity can be achieved when at least a threshold number (i.e. 1, 2, 3, 6, 11, etc.) or percentage (i.e. 55%, 61%, 78%, 82%, 99%, etc.) of Object Properties 630 or portions thereof from the compared Object Representations 625 match or substantially match. Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching Object Properties 630 or portions thereof from the compared Object Representations 625 exceeds a threshold number (i.e. 1, 2, 3, 6, 11, etc.) or a threshold percentage (i.e. 55%, 61%, 78%, 82%, 99%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or percentage of Object Properties 630 or portions thereof from the compared Object Representations 625 match or substantially match. Such thresholds can be defined by a user, by ACAAO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In further aspects, Similarity Comparison 125 can utilize Categories 635 associated with Object Properties 630 for determining substantial similarity of Object Representations 625. In one example, Object Properties 630 or portions thereof from the compared Object Representations 625 in a same Category 635 may be compared. This way, Object Properties 630 or portions thereof can be compared with their own peers. In one instance, Object Properties 630 or portions thereof from the compared Object Representations 625 in Category

635 "Type" may be compared. Any text comparison technique can be utilized in such comparing. In another instance, Object Properties 630 or portions thereof from the compared Object Representations 625 in Category 635 "Distance" or "Bearing" may be compared. Any number comparison technique can be utilized in such comparing. In a further instance, Object Properties 630 or portions thereof from the compared Object Representations 625 in Category 635 "Shape" may be compared. Any model or other computer construct comparison technique can be utilized in such comparing. In further aspects, Similarity Comparison 125 can utilize the importance (i.e. as indicated by importance index [later described], etc.) of Object Properties 630 or portions thereof for determining substantial similarity of Object Representations 625. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to more important Object Properties 630 or portions thereof such as Object Properties 630 or portions thereof in Categories 635 "Type", "Distance", "Bearing", etc., thereby tolerating mismatches in less important Object Properties 630 or portions thereof such as Object Properties 630 or portions thereof in Categories 635 "Identity", "Shape", etc. In general, any Object Property 630 or portion thereof can be assigned higher or lower importance. In further aspects, Similarity Comparison 125 can omit some of the Object Properties 630 or portions thereof from the comparison in determining substantial similarity of Object Representations 625. In one example, Object Properties 630 or portions thereof in Category 635 "Identity" can be omitted from comparison. In another example, Object Properties 630 or portions thereof in Category 635 "Shape" can be omitted from comparison. In general, any Object Property 630 or portion thereof can be omitted from comparison depending on implementation.

Similarity Comparison 125 can automatically adjust (i.e. increase or decrease) the strictness of the rules for determining substantial similarity of Object Representations 625. In some aspects, such adjustment in strictness can be done by Similarity Comparison 125 in response to determining that total equivalence of compared Object Representations 625 had not been found. Similarity Comparison 125 can keep adjusting the strictness rules until a substantial similarity is found. All the rules or settings of substantial similarity can be set, reset, or adjusted by Similarity Comparison 125 in response to another strictness level determination. For example, Similarity Comparison 125 may attempt to find a match or substantial match in a certain percentage (i.e. 90%, etc.) of Object Properties 630 or portions thereof from the compared Object Representations 625. If the comparison does not determine substantial similarity of compared Object Representations 625, Similarity Comparison 125 may decide to decrease the strictness of the rules. In response, Similarity Comparison 125 may attempt to find fewer matching or substantially matching Object Properties 630 or portions thereof than in the previous attempt using stricter rules. If the comparison still does not determine substantial similarity of compared Object Representations 625, Similarity Comparison 125 may further decrease the strictness (i.e. down to a certain minimum strictness or threshold, etc.) by requiring fewer Object Properties 630 or portions thereof to match or substantially match, thereby further increasing a chance of finding substantial similarity in compared Object Representations 625. In further aspects, an adjustment in strictness can be done by Similarity Comparison 125 in response to determining that multiple substantially similar Object Representations 625 had been found. Similarity Comparison 125 can keep adjusting the strictness of the rules until a best of the substantially similar Object Representations 625 is found. For example, Similarity Comparison 125 may attempt to find a match or substantial match in a certain percentage (i.e. 67%, etc.) of Object Properties 630 or portions thereof from the compared Object Representations 625. If the comparison determines a number of substantially similar Object Representations 625, Similarity Comparison 125 may decide to increase the

strictness of the rules to decrease the number of substantially similar Object Representations 625. In response, Similarity Comparison 125 may attempt to find more matching or substantially matching Object Properties 630 or portions thereof in addition to the earlier found Object Properties 630 or portions thereof to limit the number of substantially similar Object Representations 625. If the comparison still provides more than one substantially similar Object Representation 625, Similarity Comparison 125 may further increase the strictness by requiring additional Object Properties 630 or portions thereof to match or substantially match, thereby further narrowing the number of substantially similar Object Representations 625 until a best substantially similar Object Representation 625 is found.

Where a reference to Object Property 630 is used herein it should be understood that a portion of Object Property 630 or a plurality of Object Properties 630 can be used instead of or in addition to the Object Property 630. In one example, instead of or in addition to Object Property 630, characters, words, numbers, and/or other portions that constitute an Object Property 630 can be compared. In another example, instead of or in addition to Object Property 630, a plurality of Object Properties 630 can be compared. As such, any operations, rules, logic, and/or functions operating on Object Property 630 may similarly apply to any portion of Object Property 630 and/or a plurality of Object Properties 630 as applicable. In general, whole Object Properties 630, portions of Object Properties 630, and/or pluralities of Object Properties 630, including any operations thereon, can be combined to arrive at desired results. Some or all of the above-described rules, logic, and/or techniques can be utilized alone or in combination with each other or with other rules, logic, and/or techniques. One of ordinary skill in art will recognize that other techniques known in art for determining similarity of Object Properties 630 and/or other data that would be too voluminous to describe are within the scope of this disclosure.

In some embodiments where compared Knowledge Cells 800 include a stream of Collections of Object Representations 525, in determining similarity of Knowledge Cells 800, Similarity Comparison 125 can perform collective comparison of Collections of Object Representations 525 or portions (i.e. Object Representations 625, Object Properties 630, etc.) thereof such as comparison of a stream of Collections of Object Representations 525 or portions thereof from one Knowledge Cell 800 with a stream of Collections of Object Representations 525 or portions thereof from another Knowledge Cell 800. Similarity Comparison 125 of collectively compared Collections of Object Representations 525 or portions thereof may include any features, functionalities, and embodiments of the previously described Similarity Comparison 125 of individually compared Collections of Object Representations 525 or portions thereof. In some aspects, total equivalence is found when all Collections of Object Representations 525 or portions thereof from one Knowledge Cell 800 match all Collections of Object Representations 525 or portions thereof from another Knowledge Cell 800. If total equivalence is not found, Similarity Comparison 125 may attempt to determine substantial or other similarity of compared Knowledge Cells 800. In one example, substantial similarity can be achieved when most of the Collections of Object Representations 525 or portions thereof from the compared Knowledge Cells 800 match or substantially match. In another example, substantial similarity can be achieved when at least a threshold number (i.e. 1, 2, 4, 9, 33, 138, etc.) or percentage (i.e. 39%, 58%, 77%, 88%, 94%, etc.) of Collections of Object Representations 525 or portions thereof from the compared Knowledge Cells 800 match or substantially match. Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching Collections of Object Representations 525 or portions thereof from the compared Knowledge Cells 800 exceeds a threshold number (i.e. 1, 2, 4, 9, 33, 138, etc.) or a threshold percentage (i.e. 39%, 58%, 77%,

88%, 94%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or percentage of Collections of Object Representations 525 or portions thereof from the compared Knowledge Cells 800 match or substantially match. Such thresholds can be defined by a user, by ACAAO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In some aspects, Similarity Comparison 125 can utilize the importance (i.e. as indicated by importance index [later described], etc.) of Collections of Object Representations 525 or portions thereof for determining substantial similarity of Knowledge Cells 800. In one example, substantial similarity can be achieved when matches or substantial matches are found with respect to more important Collections of Object Representations 525 or portions thereof such as more substantive or larger Collections of Object Representations 525 (i.e. Collections of Object Representations 525 comprising a higher number of Object Representations 625, etc.) or portions thereof, etc., thereby tolerating mismatches in less important Collections of Object Representations 525 or portions thereof such as less substantive or smaller Collections of Object Representations 525 (i.e. Collections of Object Representations 525 comprising a lower number of Object Representations 625, etc.) or portions thereof, etc. In general, any Collection of Object Representations 525 or portion thereof can be assigned higher or lower importance. In other aspects, Similarity Comparison 125 can utilize the order of Collections of Object Representations 525 or portions thereof for determining substantial similarity of Knowledge Cells 800. In one example, substantial similarity can be achieved when matches or substantial matches are found in earlier Collections of Object Representations 525 or portions thereof from the compared Knowledge Cells 800, thereby tolerating mismatches in later Collections of Object Representations 525 or portions thereof. In another example, substantial similarity can be achieved when matches or substantial matches are found in corresponding (i.e. similarly ordered, temporally related, etc.) Collections of Object Representations 525 or portions thereof from the compared Knowledge Cells 800. In one instance, a 86th Collection of Object Representations 525 or portions thereof from one Knowledge Cell 800 can be compared with a 86th Collection of Object Representations 525 or portions thereof from another Knowledge Cell 800. In another instance, a 86th Collection of Object Representations 525 or portions thereof from one Knowledge Cell 800 can be compared with a number of Collections of Object Representations 525 or portions thereof around (i.e. preceding and/or following) a 86th Collection of Object Representations 525 from another Knowledge Cell 800. This way, flexibility can be implemented in finding a substantially similar Collection of Object Representations 525 or portions thereof if the Collections of Object Representations 525 or portions thereof in the compared Knowledge Cells 800 are not perfectly aligned. In a further instance, Similarity Comparison 125 can utilize Dynamic Time Warping (DTW) and/or other techniques known in art for comparing and/or aligning temporal sequences (i.e. streams of Collections of Object Representations 525 or portions thereof, etc.) that may vary in time or speed. In further aspects, Similarity Comparison 125 can omit some of the Collections of Object Representations 525 or portions thereof from the comparison in determining substantial similarity of Knowledge Cells 800. In one example, less substantive or smaller Collections of Object Representations 525 or portions thereof can be omitted from comparison. In another example, some or all Collections of Object Representations 525 or portions thereof related to a specific time period can be omitted from comparison. In general, any Collection of Object Representations 525 or portion thereof can be omitted from comparison depending on implementation.

Similarity Comparison 125 can automatically adjust (i.e. increase or decrease) the strictness of the rules for determining substantial similarity of Knowledge Cells 800. In some aspects, such adjustment in strictness can be

done by Similarity Comparison 125 in response to determining that total equivalence of compared Knowledge Cells 800 had not been found. Similarity Comparison 125 can keep adjusting the strictness of the rules until substantial similarity is found. All the rules or settings of substantial similarity can be set, reset, or adjusted by Similarity Comparison 125 in response to another strictness level determination. For example, Similarity Comparison 125 may attempt to find a match or substantial match in a certain percentage (i.e. 89%, etc.) of Collections of Object Representations 525 or portions thereof from the compared Knowledge Cells 800. If the comparison does not determine substantial similarity of compared Knowledge Cells 800, Similarity Comparison 125 may decide to decrease the strictness of the rules. In response, Similarity Comparison 125 may attempt to find fewer matching or substantially matching Collections of Object Representations 525 or portions thereof than in the previous attempt using stricter rules. If the comparison still does not determine substantial similarity of compared Knowledge Cells 800, Similarity Comparison 125 may further decrease (i.e. down to a certain minimum strictness or threshold, etc.) the strictness by requiring fewer Collections of Object Representations 525 or portions thereof to match or substantially match, thereby further increasing a chance of finding substantial similarity in compared Knowledge Cells 800. In further aspects, an adjustment in strictness can be done by Similarity Comparison 125 in response to determining that multiple substantially similar Knowledge Cells 800 had been found. Similarity Comparison 125 can keep adjusting the strictness of the rules until a best of the substantially similar Knowledge Cells 800 is found. For example, Similarity Comparison 125 may attempt to find a match or substantial match in a certain percentage (i.e. 69%, etc.) of Collections of Object Representations 525 or portions thereof from the compared Knowledge Cells 800. If the comparison determines a number of substantially similar Knowledge Cells 800, Similarity Comparison 125 may decide to increase the strictness of the rules to decrease the number of substantially similar Knowledge Cells 800. In response, Similarity Comparison 125 may attempt to find more matching or substantially matching Collections of Object Representations 525 or portions thereof in addition to the earlier found Collections of Object Representations 525 or portions thereof to limit the number of substantially similar Knowledge Cells 800. If the comparison still provides more than one substantially similar Knowledge Cell 800, Similarity Comparison 125 may further increase the strictness by requiring additional Collections of Object Representations 525 or portions thereof to match or substantially match, thereby further narrowing the number of substantially similar Knowledge Cells 800 until a best substantially similar Knowledge Cell 800 is found.

Some or all of the aforementioned rules, logic, and/or techniques for determining substantial similarity of Knowledge Cells 800 can be utilized alone or in combination with each other or with other rules, logic, and/or techniques. One of ordinary skill in art will recognize that other techniques known in art for determining similarity of Knowledge Cells 800 and/or other data that would be too voluminous to describe are within the scope of this disclosure.

In any of the comparisons involving numbers such as, for example, Object Properties 630 including numbers (i.e. distances, bearings/angles, etc.), Similarity Comparison 125 can compare a number from one Object Property 630 with a number from another Object Property 630. In some aspects, total equivalence is found when the number from one Object Property 630 equals the number from another Object Property 630. In other aspects, if total equality is not found, Similarity Comparison 125 may attempt to determine substantial similarity of the compared numbers using a tolerance or threshold for determining a match. In some aspects, Similarity Comparison 125 can utilize a threshold for acceptable number difference in determining a match of compared numbers. For example, a

threshold for acceptable number difference (i.e. absolute difference, etc.) can be set at 10. Therefore, 130 matches or is sufficiently similar to 135 because the number difference (i.e. 5 in this example) is lower than the threshold for acceptable number difference (i.e. 10 in this example, etc.). Furthermore, 130 does not match or is not sufficiently similar to 143 because the number difference (i.e. 13 in this example) is greater than the threshold for acceptable number difference. Any other threshold for acceptable number difference can be used such as 0.024, 1, 8, 15, 77, 197, 2438, 728322, and/or others. In other aspects, Similarity Comparison 125 can utilize a threshold for acceptable percentage difference in determining a match of compared numbers. For example, a threshold for acceptable percentage difference can be set at 10%. Therefore, 100 matches or is sufficiently similar to 106 because the percentage difference (i.e. 6% in this example) is lower than the threshold for acceptable percentage difference (i.e. 10% in this example). Furthermore, 100 does not match or is not sufficiently similar to 84 because the percentage difference (i.e. 16% in this example) is higher than the threshold for acceptable percentage difference. Any other threshold for acceptable percentage difference can be used such as 0.68%, 1%, 3%, 11%, 33%, 69%, 87%, and/or others. The aforementioned thresholds can be defined by a user, by ACAAO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. Other techniques known in art for comparing numbers can be utilized herein.

In any of the comparisons involving text such as, for example, Object Properties 630 including text (i.e. types, identities, etc.), Similarity Comparison 125 can compare words, characters, and/or other text from one Object Property 630 with words, characters, and/or other text from another Object Property 630. In some aspects, total equivalence is found when all words, characters, and/or other text from one Object Property 630 match all words, characters, and/or other text from another Object Property 630. In other aspects, if total equivalence is not found, Similarity Comparison 125 may attempt to determine substantial similarity of compared Object Properties 630. In one example, substantial similarity can be achieved when most of the words, characters, and/or other text from the compared Object Properties 630 match or substantially match. In another example, substantial similarity can be achieved when at least a threshold number (i.e. 1, 2, 3, 4, 7, 11, etc.) or percentage (i.e. 38%, 63%, 77%, 84%, 98%, etc.) of words, characters, and/or other text from the compared Object Properties 630 match or substantially match. Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching words, characters, and/or other text from the compared Object Properties 630 exceeds a threshold number (i.e. 1, 2, 3, 4, 7, 11, etc.) or a threshold percentage (i.e. 48%, 63%, 77%, 84%, 98%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or percentage of words, characters, and/or other text from the compared Object Properties 630 match or substantially match. Such thresholds can be defined by a user, by ACAAO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In further aspects, Similarity Comparison 125 can utilize the importance (i.e. as indicated by importance index [later described], etc.) of words, characters, and/or other text for determining substantial similarity of Object Properties 630. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to more important words, characters, and/or other text such as longer words and/or other text, thereby tolerating mismatches in less important words, characters, and/or other text such as shorter words and/or other text. In general, any word, character, and/or other text can be assigned higher or lower importance. In further aspects, Similarity Comparison 125 can utilize the order of words, characters, and/or other text for determining substantial similarity of Object Properties 630. For example,

substantial similarity can be achieved when matches or substantial matches are found with respect to front-most words, characters, and/or other text, thereby tolerating mismatches in later words, characters, and/or other text. In further aspects, Similarity Comparison 125 can utilize semantic conversion to account for variations of words and/or other text. In one example, Object Property 630 may include a word "house". In addition to searching for the exact word in a compared Object Property 630, Similarity Comparison 125 can employ semantic conversion and attempt to match "home", "residence", "dwelling", "place", or other semantically similar variations of the word with a meaning "house". In another example, Object Property 630 may include a word "buy". In addition to searching for the exact word in a compared Object Property 630, Similarity Comparison 125 can employ semantic conversion and attempt to match "buying", "bought", or other semantically similar variations of the word with a meaning "buy" in different tenses. Any other grammatical analysis or transformation can be utilized to cover the full scope of word and/or other text variations. In some designs, semantic conversion can be implemented using a thesaurus or dictionary. In another example, semantic conversion can be implemented using a table where each row comprises semantically similar variations of a word and/or other text. In further aspects, Similarity Comparison 125 can utilize a language model for understanding or interpreting the concepts contained in the words and/or other text and compare the concepts instead of or in addition to the words and/or other text. Examples of language models include unigram model, n-gram model, neural network language model, bag of words model, and/or others. Any of the techniques for matching of words can similarly be used for matching of concepts. In further aspects, Similarity Comparison 125 can omit some of the words, characters, and/or other text from the comparison in determining substantial similarity of Object Properties 630. In one example, rear-most words, characters, and/or other text can be omitted from comparison. In another example, shorter words and/or other text can be omitted from comparison. In general, any word, character, and/or other text can be omitted from comparison depending on implementation. Other techniques known in art for comparing words, characters, and/or other text can be utilized herein.

In some embodiments, Similarity Comparison 125 can compare one or more Extra Info 527 (i.e. time information, location information, computed information, visual information, acoustic information, contextual information, and/or other information, etc.) in addition to or instead of comparing Collections of Object Representations 525 or portions thereof in determining substantial similarity of Knowledge Cells 800. Extra Info 527 can be set to be less, equally, or more important (i.e. as indicated by importance index [later described], etc.) than Collections of Object Representations 525, Object Representations 625, Object Properties 630, and/or other elements in the comparison. Since Extra Info 527 may include any contextual or other information that can be useful in determining similarity of any compared elements, Extra Info 527 can be used to enhance any of the aforementioned similarity determinations as applicable.

In some embodiments, Similarity Comparison 125 can also compare one or more Instruction Sets 526 in addition to or instead of comparing Collections of Object Representations 525 or portions thereof in determining substantial similarity of Knowledge Cells 800. In some aspects, Similarity Comparison 125 can compare portions of Instruction Sets 526 to determine substantial or other similarity of Instruction Sets 526. Similar to the above-described thresholds, thresholds for the number or percentage of matching portions of the compared Instruction Sets 526 can be utilized in determining substantial or other similarity of the compared Instruction Sets 526. Such thresholds can be defined by a user, by ACAAO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In other aspects,

Similarity Comparison 125 can compare text (i.e. characters, words, phrases, etc.), numbers, or other data (i.e. bits, etc.) to determine substantial or other similarity of Instruction Sets 526. Any other comparison technique can be utilized in comparing Instruction Sets 526 in alternate implementations. Instruction Sets 526 can be set to be less, equally, or more important (i.e. as indicated by importance index [later described], etc.) than Collections of Object Representations 525, Object Representations 625, Object Properties 630, Extra Info 527, and/or other elements in the comparison.

In some embodiments, an importance index (not shown) or other importance ranking technique can be used in any of the previously described comparisons or other processing involving elements of different importance. Importance index indicates importance of the element to or with which the index is assigned or associated. For example, importance index may indicate importance of a Knowledge Cell 800, Collection of Object Representations 525, Object Representation 625, Object Property 630, Instruction Set 526, Extra Info 527, and/or other element to or with which the index is assigned or associated. In some aspects, importance index on a scale from 0 to 1 can be utilized, although, any other range can also be utilized. Importance index can be stored in or associated with the element to which the index pertains. Importance indexes of various elements can be defined by a user, by ACAAO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. In one example, a higher Importance index can be assigned to more substantive or larger Collections of Object Representations 525 (i.e. Collections of Object Representations 525 comprising a higher number of Object Representations 625, etc.). In another example, a higher importance index can be assigned to Object Representations 625 representing closer, larger, and/or other Objects 615. Any importance index can be assigned to or associated with any element described herein depending on implementation. Any importance ranking technique can be utilized as or instead of importance index in alternate embodiments.

In some embodiments, Similarity Comparison 125 may generate a similarity index (not shown) for any of the compared elements. Similarity index indicates how well an element is matched with another element. For example, similarity index indicates how well a Knowledge Cell 800, Collection of Object Representations 525, Object Representation 625, Object Property 630, Instruction Set 526, Extra Info 527, and/or other element is matched with a compared element. In some aspects, similarity index on a scale from 0 to 1 can be utilized, although, any other range can also be utilized. Similarity index can be generated by Similarity Comparison 125 whether substantial or other similarity between the compared elements is achieved or not. In one example, similarity index can be determined for a Knowledge Cell 800 based on a ratio/percentage of matched or substantially matched Collections of Object Representations 525 relative to the number of Collections of Object Representations 525 in the compared Knowledge Cell 800. Specifically, similarity index of 0.89 is determined if 89% of Collections of Object Representations 525 of one Knowledge Cell 800 match or substantially match Collections of Object Representations 525 of another Knowledge Cell 800. In some designs, importance (i.e. as indicated by importance index, etc.) of one or more Collections of Object Representations 525 can be included in the calculation of a weighted similarity index. Similar determination of similarity index can be implemented with Collections of Object Representations 525, Object Representations 625, Object Properties 630, Instruction Sets 526, Extra Info 527, and/or other elements or portions thereof. Any combination of the aforementioned similarity index determinations or calculations can be utilized in

alternate embodiments. Any similarity ranking technique can be utilized to determine or calculate similarity index in alternate embodiments.

Referring to Fig. 20, an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Neural Network 530a comprising shortcut Connections 853 is illustrated. In some designs, Knowledge Cells 800 in one Layer 854 of Neural Network 530a can be connected with Knowledge Cells 800 in any Layer 854, not only in a successive Layer 854, thereby creating shortcuts (i.e. shortcut Connections 853, etc.) through Neural Network 530a. In some aspects, creating a shortcut Connection 853 can be implemented by performing Similarity Comparisons 125 of a Knowledge Cell 800 from Knowledge Structuring Unit 520 with Knowledge Cells 800 in any Layer 854 when applying (i.e. storing, copying, etc.) the Knowledge Cell 800 from Knowledge Structuring Unit 520 onto Neural Network 530a. Once created, shortcut Connections 853 enable a wider variety of Knowledge Cells 800 to be considered when selecting a path through Neural Network 530a. In some embodiments, Knowledge Structuring Unit 520 structures or generates Knowledge Cells 800 and the system applies them onto Neural Network 530a, thereby implementing learning Avatar's 605 operation in circumstances including objects with various properties. The system can perform Similarity Comparisons 125 of a Knowledge Cell 800 from Knowledge Structuring Unit 520 with Knowledge Cells 800 in one or more Layers 854 of Neural Network 530a. If a substantially similar Knowledge Cell 800 is not found in the one or more Layers 854 of Neural Network 530a, the system may insert (i.e. copy, store, etc.) the Knowledge Cell 800 from Knowledge Structuring Unit 520 into a Layer 854 of Neural Network 530a, and create a Connection 853 to the inserted Knowledge Cell 800 from a prior Knowledge Cell 800 including assigning an occurrence count to the new Connection 853, calculating a weight of the new Connection 853, and updating any other Connections 853 originating from the prior Knowledge Cell 800. On the other hand, if a substantially similar Knowledge Cell 800 is found in the one or more Layers 854 of Neural Network 530a, the system may update occurrence count and weight of Connection 853 to that Knowledge Cell 800 from a prior Knowledge Cell 800, and update any other Connections 853 originating from the prior Knowledge Cell 800. Any of the previously described and/or other techniques for comparing, inserting, updating, and/or other operations on Knowledge Cells 800, Connections 853, Layers 854, and/or other elements can similarly be utilized in Neural Network 530a that comprises shortcut Connections 853.

Referring to Fig. 21, an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Graph 530b is illustrated. In some aspects, any Knowledge Cell 800 can be connected with any other Knowledge Cell 800 in Graph 530b. In other aspects, any Knowledge Cell 800 can be connected with itself and/or any other Knowledge Cell 800 in Graph 530b. In some embodiments, Knowledge Structuring Unit 520 structures or generates Knowledge Cells 800 and the system applies (i.e. store, copy, etc.) them onto Graph 530b, thereby implementing learning Avatar's 605 operation in circumstances including objects with various properties. The system can perform Similarity Comparisons 125 of a Knowledge Cell 800 from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Graph 530b. If a substantially similar Knowledge Cell 800 is not found in Graph 530b, the system may insert (i.e. copy, store, etc.) the Knowledge Cell 800 from Knowledge Structuring Unit 520 into Graph 530b, and create a Connection 853 to the inserted Knowledge Cell 800 from a prior Knowledge Cell 800 including assigning an occurrence count to the new Connection 853, calculating a weight of the new Connection 853, and updating any other Connections 853 originating from the prior Knowledge Cell 800. On the other hand, if a substantially similar Knowledge Cell 800 is

found in Graph 530b, the system may update occurrence count and weight of Connection 853 to that Knowledge Cell 800 from a prior Knowledge Cell 800, and update any other Connections 853 originating from the prior Knowledge Cell 800. Any of the previously described and/or other techniques for comparing, inserting, updating, and/or other operations on Knowledge Cells 800, Connections 853, and/or other elements can similarly be utilized in Graph 530b.

For example, the system can perform Similarity Comparisons 125 of Knowledge Cell 800ba from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Graph 530b. In the case that a substantially similar match is not found, the system may insert Knowledge Cell 800ha into Graph 530b and copy Knowledge Cell 800ba into the inserted Knowledge Cell 800ha. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bb from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Graph 530b. In the case that a substantially similar match is found between Knowledge Cell 800bb and Knowledge Cell 800hb, the system may create Connection 853h1 between Knowledge Cell 800ha and Knowledge Cell 800hb with occurrence count of 1 and weight of 1. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bc from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Graph 530b. In the case that a substantially similar match is found between Knowledge Cell 800bc and Knowledge Cell 800hc, the system may update occurrence count and weight of Connection 853h2 between Knowledge Cell 800hb and Knowledge Cell 800hc, and update weights of other outgoing Connections 853 (one in this example) originating from Knowledge Cell 800hb as previously described. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bd from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Graph 530b. In the case that a substantially similar match is not found, the system may insert Knowledge Cell 800hd into Graph 530b and copy Knowledge Cell 800bd into the inserted Knowledge Cell 800hd. The system may also create Connection 853h3 between Knowledge Cell 800hc and Knowledge Cell 800hd with occurrence count of 1 and weight calculated based on the occurrence count as previously described. The system may also update weights of other outgoing Connections 853 (one in this example) originating from Knowledge Cell 800hc as previously described. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800be from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Graph 530b. In the case that a substantially similar match is not found, the system may insert Knowledge Cell 800he into Graph 530b and copy Knowledge Cell 800be into the inserted Knowledge Cell 800he. The system may also create Connection 853h4 between Knowledge Cell 800hd and Knowledge Cell 800he with occurrence count of 1 and weight of 1. Applying any additional Knowledge Cells 800 from Knowledge Structuring Unit 520 onto Graph 530b follows similar logic or process as the above-described.

Referring to Fig. 22, an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Collection of Sequences 530c is illustrated. Collection of Sequences 530c comprises the functionality for storing one or more Sequences 533. Sequence 533 comprises the functionality for storing any number of Knowledge Cells 800. For example, Knowledge Structuring Unit 520 structures or generates Knowledge Cells 800 and the system applies them onto Collection of Sequences 530c, thereby implementing learning Avatar's 605 operation in circumstances including objects with various properties. The system can perform collective Similarity Comparisons 125 of Knowledge Cells 800 from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Sequences 533 of Collection of Sequences 530c to find a Sequence 533 comprising Knowledge Cells 800 that are collectively

substantially similar to the Knowledge Cells 800 from Knowledge Structuring Unit 520. If Sequence 533 comprising such collectively substantially similar Knowledge Cells 800 is not found in Collection of Sequences 530c, the system may create a new Sequence 533 comprising the Knowledge Cells 800 from Knowledge Structuring Unit 520 and insert (i.e. copy, store, etc.) the new Sequence 533 into Collection of Sequences 530c. On the other hand, if Sequence 533 comprising collectively substantially similar Knowledge Cells 800 is found in Collection of Sequences 530c, the system may optionally omit inserting the Knowledge Cells 800 from Knowledge Structuring Unit 520 into Collection of Sequences 530c as inserting a similar Sequence 533 may not add much or any additional knowledge. This approach can save storage resources and limit the number of Knowledge Cells 800 that may later need to be processed or compared. In some aspects, a Sequence 533 may include Knowledge Cells 800 relating to a single operation of Avatar 605. In other aspects, a Sequence 533 may include Knowledge Cells 800 relating to a part of an operation of Avatar 605. In further aspects, one or more long Sequences 533 each including Knowledge Cells 800 of multiple operations of Avatar 605 can be utilized. In one example, Knowledge Cells 800 of all operations can be stored in a single long Sequence 533 in which case Collection of Sequences 530c as a separate element can be omitted. In another example, Knowledge Cells 800 of multiple operations can be included in a plurality of long Sequences 533 such as hourly, daily, weekly, monthly, yearly, or other periodic or other Sequences 533. Similarity Comparisons 125 can be performed by traversing the one or more long Sequences 533 to find a match or substantially similar match. For instance, the system can perform collective Similarity Comparisons 125 of Knowledge Cells 800 from Knowledge Structuring Unit 520 with Knowledge Cells 800 in subsequences of a long Sequence 533 in incremental or other traversing pattern to find a subsequence comprising Knowledge Cells 800 that are collectively substantially similar to the Knowledge Cells 800 from Knowledge Structuring Unit 520. The incremental traversing pattern may start from one end of a long Sequence 533 and move the comparison subsequence up or down one or any number of incremental Knowledge Cells 800 at a time. Other traversing patterns or methods can be employed such as starting from the middle of the Sequence 533 and subdividing the resulting sub-sequences in a recursive pattern, or any other traversing pattern or method. If a subsequence comprising collectively substantially similar Knowledge Cells 800 is not found in the long Sequence 533, the system may concatenate or append the Knowledge Cells 800 from Knowledge Structuring Unit 520 to the long Sequence 533. In further aspects, Connections 853 can optionally be used in Sequence 533 to connect Knowledge Cells 800. For example, a Knowledge Cell 800 can be connected not only with a next Knowledge Cell 800 in the Sequence 533, but also with any other Knowledge Cell 800 in the Sequence 533, thereby creating alternate routes or shortcuts through the Sequence 533. Any number of Connections 853 connecting any Knowledge Cells 800 can be utilized. Any of the previously described and/or other techniques for comparing, inserting, updating, and/or other operations on Knowledge Cells 800, Connections 853, and/or other elements can similarly be utilized in Sequences 533 and/or Collection of Sequences 530c.

In some embodiments, various elements and/or techniques can be utilized in the aforementioned substantial similarity determinations with respect to collectively compared Knowledge Cells 800 and/or other elements. In some aspects, substantial similarity of collectively compared Knowledge Cells 800 can be determined based on similarities or similarity indexes of the individually compared Knowledge Cells 800. In one example, an average of similarities or similarity indexes of individually compared Knowledge Cells 800 can be used to determine similarity of collectively compared Knowledge Cells 800. In another example, a weighted average of similarities or

similarity indexes of individually compared Knowledge Cells 800 can be used to determine similarity of collectively compared Knowledge Cells 800. For instance, to affect the weighting of collective similarity, a higher weight or importance (i.e. importance index, etc.) can be assigned to the similarities or similarity indexes of some Knowledge Cells 800 and lower for other Knowledge Cells 800. Any higher or lower weight or importance assignment can be implemented. In other aspects, any of the previously described or other thresholds for substantial similarity of individually compared elements can similarly be utilized for collectively compared elements. In one example, substantial similarity of collectively compared Knowledge Cells 800 can be achieved when their collective similarity or similarity index exceeds a similarity threshold. In another example, substantial similarity of collectively compared Knowledge Cells 800 can be achieved when at least a threshold number or percentage of Knowledge Cells 800 from the collectively compared Knowledge Cells 800 match or substantially match. Similarly, substantial similarity of collectively compared Knowledge Cells 800 can be achieved when a number or percentage of matching or substantially matching Knowledge Cells 800 from the collectively compared Knowledge Cells 800 exceeds a threshold. Such thresholds can be defined by a user, by ACAAO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. Similar elements and/or techniques as the aforementioned can be used for similarity determinations of other collectively compared elements such as Collections of Object Representations 525, Object Representations 625, Object Properties 630, Instruction Sets 526, Extra Info 527, and/or others. Similarity determinations of collectively compared elements may include any features, functionalities, and embodiments of Similarity Comparison 125, and vice versa.

Any of the previously described data structures or arrangements of Knowledge Cells 800 such as Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, and/or others can be used alone, or in combination with each other or with other elements, in alternate embodiments. In one example, a path in Neural Network 530a or Graph 530b may include its own separate sequence of Knowledge Cells 800 that are not interconnected with Knowledge Cells 800 in other paths. In another example, a part of a path in Neural Network 530a or Graph 530b may include a sequence of Knowledge Cells 800 interconnected with Knowledge Cells 800 in other paths, whereas, another part of the path may include its own separate sequence of Knowledge Cells 800 that are not interconnected with Knowledge Cells 800 in other paths. Any other combinations or arrangements of Knowledge Cells 800 can be implemented.

Referring to Fig. 23, an embodiment of determining anticipatory Instruction Sets 526 from a single Knowledge Cell 800 is illustrated. Knowledge Cell 800 may be part of a Knowledgebase 530 (i.e. Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.) such as Collection of Knowledge Cells 530d. Decision-making Unit 540 comprises the functionality for anticipating or determining Avatar's 605 operation in circumstances including objects with various properties. Decision-making Unit 540 comprises the functionality for anticipating or determining Instruction Sets 526 to be used or executed in Avatar's 605 autonomous operation. In some aspects, Instruction Sets 526 anticipated or determined to be used or executed in Avatar's 605 autonomous operation may be referred to as anticipatory Instruction Sets 526, alternate Instruction Sets 526, and/or other suitable name or reference. Therefore, these terms can be used interchangeably herein depending on context. Decision-making Unit 540 also comprises other disclosed functionalities.

In some aspects, Decision-making Unit 540 may anticipate or determine Instruction Sets 526 (i.e. anticipatory Instruction Sets 526, etc.) for autonomous Avatar 605 operation by performing Similarity Comparisons

125 of incoming Collections of Object Representations 525 or portions thereof from Object Processing Unit 140 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Knowledgebase 530 (i.e. Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.). A Knowledge Cell 800 includes knowledge (i.e. one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527, etc.) of how Avatar 605 operated in a circumstance including objects with various properties as previously described. When one or more Collections of Object Representations 525 representing objects with similar properties are received in the future, Decision-making Unit 540 can anticipate the Instruction Sets 526 (i.e. anticipatory Instruction Sets 526, etc.) previously learned in a similar circumstance, thereby enabling autonomous Avatar 605 operation. In some aspects, Decision-making Unit 540 can perform Similarity Comparisons 125 of incoming Collections of Object Representations 525 from Object Processing Unit 140 with Collections of Object Representations 525 from Knowledge Cells 800 in Knowledgebase 530 (i.e. Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.). If one or more substantially similar Collections of Object Representations 525 or portions thereof are found in a Knowledge Cell 800 from Knowledgebase 530, Instruction Sets 526 (i.e. anticipatory Instruction Sets 526, etc.) for autonomous Avatar 605 operation can be anticipated in Instruction Sets 526 correlated with the one or more Collections of Object Representations 525 from the Knowledge Cell 800. In some designs, subsequent one or more Instruction Sets 526 for autonomous Avatar 605 operation can be anticipated in Instruction Sets 526 correlated with subsequent Collections of Object Representations 525 from the Knowledge Cell 800 or other Knowledge Cells 800, thereby anticipating not only current, but also additional future Instruction Sets 526. Although, Extra Info 527 is not shown in this and/or other figures for clarity of illustration, it should be noted that any Collection of Object Representations 525, Instruction Set 526, and/or other element may include or be associated with Extra Info 527 and that Decision-making Unit 540 can utilize Extra Info 527 for enhanced decision making.

For example, Decision-making Unit 540 can perform Similarity Comparison 125 of Collection of Object Representations 525l1 or portions thereof from Object Processing Unit 140 with Collection of Object Representations 525a1 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a1 or portions thereof from Knowledge Cell 800oa may be found substantially similar. Decision-making Unit 540 can anticipate Instruction Sets 526a1-526a3 correlated with Collection of Object Representations 525a1, thereby enabling autonomous Avatar 605 operation. Decision-making Unit 540 can then perform Similarity Comparison 125 of Collection of Object Representations 525l2 or portions thereof from Object Processing Unit 140 with Collection of Object Representations 525a2 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a2 or portions thereof from Knowledge Cell 800oa may be found substantially similar. Decision-making Unit 540 can anticipate Instruction Set 526a4 correlated with Collection of Object Representations 525a2, thereby enabling autonomous Avatar 605 operation. Decision-making Unit 540 can then perform Similarity Comparison 125 of Collection of Object Representations 525l3 or portions thereof from Object Processing Unit 140 with Collection of Object Representations 525a3 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a3 or portions thereof from Knowledge Cell 800oa may be found substantially similar. Decision-making Unit 540 may not anticipate any Instruction Sets 526 since none are correlated with Collection of Object Representations 525a3. Decision-making Unit 540 can then perform Similarity Comparison 125 of Collection of Object Representations 525l4 or portions thereof from Object Processing Unit 140 with Collection of Object

Representations 525a4 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a4 or portions thereof from Knowledge Cell 800oa may not be found substantially similar. Decision-making Unit 540 can then perform Similarity Comparison 125 of Collection of Object Representations 525l5 or portions thereof from Object Processing Unit 140 with Collection of Object Representations 525a5 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a5 or portions thereof from Knowledge Cell 800oa may not be found substantially similar. Decision-making Unit 540 can implement similar logic or process for any additional Collections of Object Representations 525 from Object Processing Unit 140, and so on.

It should be understood that any of the described elements and/or techniques in the foregoing example can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells 800 or elements (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof would be affected accordingly. In one example, Extra Info 527 can be included in the Similarity Comparisons 125 as previously described. In another example, as history of incoming Collections of Object Representations 525 becomes available, Decision-making Unit 540 can perform collective Similarity Comparisons 125 of the history of Collections of Object Representations 525 or portions thereof from Object Processing Unit 140 with subsequences of Collections of Object Representations 525 or portions thereof from Knowledge Cell 800. In a further example, the described comparisons in a single Knowledge Cell 800 may be performed on any number of Knowledge Cells 800 sequentially or in parallel. Parallel processors such as a plurality of Processors 11 or cores thereof can be utilized for such parallel processing. In a further example, various arrangements of Collections of Object Representations 525 and/or other elements in a Knowledge Cell 800 can be utilized as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to Fig. 24, an embodiment of determining anticipatory Instruction Sets 526 by traversing a single Knowledge Cell 800 is illustrated. Knowledge Cell 800 may be part of a Knowledgebase 530 (i.e. Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.) such as Collection of Knowledge Cells 530d. For example, Decision-making Unit 540 can perform Similarity Comparison 125 of Collection of Object Representations 525l1 or portions thereof from Object Processing Unit 140 with Collection of Object Representations 525a1 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a1 or portions thereof from Knowledge Cell 800oa may not be found substantially similar. Decision-making Unit 540 can then perform Similarity Comparison 125 of Collection of Object Representations 525l1 or portions thereof from Object Processing Unit 140 with Collection of Object Representations 525a2 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a2 or portions thereof from Knowledge Cell 800oa may not be found substantially similar. Decision-making Unit 540 can then perform Similarity Comparison 125 of Collection of Object Representations 525l1 or portions thereof from Object Processing Unit 140 with Collection of Object Representations 525a3 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a3 or portions thereof from Knowledge Cell 800oa may be found substantially similar. Decision-making Unit 540 may not anticipate any Instruction Sets 526 since none are correlated with Collection of Object Representations 525a3. Decision-making Unit 540 can then perform Similarity Comparison 125 of Collection of Object Representations 525l2 or portions thereof from Object Processing Unit 140 with Collection of Object

Representations 525a4 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a4 or portions thereof from Knowledge Cell 800oa may be found substantially similar. Decision-making Unit 540 can anticipate Instruction Sets 526a5-526a6 correlated with Collection of Object Representations 525a4, thereby enabling autonomous Avatar 605 operation. Decision-making Unit 540 can then perform Similarity Comparison 125 of Collection of Object Representations 525l3 or portions thereof from Object Processing Unit 140 with Collection of Object Representations 525a5 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a5 or portions thereof from Knowledge Cell 800oa may be found substantially similar. Decision-making Unit 540 may not anticipate any Instruction Sets 526 since none are correlated with Collection of Object Representations 525a5. Decision-making Unit 540 can implement similar logic or process for any additional Collections of Object Representations 525 from Object Processing Unit 140, and so on.

It should be understood that any of the described elements and/or techniques in the foregoing example can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells 800 or elements (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof would be affected accordingly. In one example, Extra Info 527 can be included in the Similarity Comparisons 125 as previously described. In another example, as history of incoming Collections of Object Representations 525 becomes available, Decision-making Unit 540 can perform collective Similarity Comparisons 125 of the history of Collections of Object Representations 525 or portions thereof from Object Processing Unit 140 with subsequences of Collections of Object Representations 525 or portions thereof from Knowledge Cell 800. In a further example, traversing may be performed in incremental traversing pattern such as starting from one end of Knowledge Cell 800 and moving the comparison subsequence up or down the list one or any number of incremental Collections of Object Representations 525 at a time. Other traversing patterns or methods can be employed such as starting from the middle of the Knowledge Cell 800 and subdividing the resulting subsequence in a recursive pattern, or any other traversing pattern or method. In a further example, the described traversing of a single Knowledge Cell 800 may be performed on any number of Knowledge Cells 800 sequentially or in parallel. Parallel processors such as a plurality of Processors 11 or cores thereof can be utilized for such parallel processing. In a further example, various arrangements of Collections of Object Representations 525 and/or other elements in a Knowledge Cell 800 can be utilized as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to Fig. 25, an embodiment of determining anticipatory Instruction Sets 526 using collective similarity comparisons is illustrated. For example, Decision-making Unit 540 can perform Similarity Comparisons 125 of Collection of Object Representations 525l1 or portions thereof from Object Processing Unit 140 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Collection of Knowledge Cells 530d. Collection of Object Representations 525c1 or portions thereof from Knowledge Cell 800rc may be found substantially similar with highest similarity. Decision-making Unit 540 can anticipate any Instruction Sets 526 (not shown) correlated with Collection of Object Representations 525c1, thereby enabling autonomous Avatar 605 operation. Decision-making Unit 540 can then perform collective Similarity Comparisons 125 of Collections of Object Representations 525l1-525l2 or portions thereof from Object Processing Unit 140 with Collections of Object

Representations 525 or portions thereof from Knowledge Cells 800 in Collection of Knowledge Cells 530d. Collections of Object Representations 525c1-525c2 or portions thereof from Knowledge Cell 800rc may be found substantially similar with highest similarity. Decision-making Unit 540 can anticipate any Instruction Sets 526 (not shown) correlated with Collection of Object Representations 525c2, thereby enabling autonomous Avatar 605 operation. Decision-making Unit 540 can then perform collective Similarity Comparisons 125 of Collections of Object Representations 525l1-525l3 or portions thereof from Object Processing Unit 140 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Collection of Knowledge Cells 530d. Collections of Object Representations 525d1-525d3 or portions thereof from Knowledge Cell 800rd may be found substantially similar with highest similarity. Decision-making Unit 540 can anticipate any Instruction Sets 526 (not shown) correlated with Collection of Object Representations 525d3, thereby enabling autonomous Avatar 605 operation. Decision-making Unit 540 can then perform collective Similarity Comparisons 125 of Collections of Object Representations 525l1-525l4 or portions thereof from Object Processing Unit 140 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Collection of Knowledge Cells 530d. Collections of Object Representations 525d1-525d4 or portions thereof from Knowledge Cell 800rd may be found substantially similar with highest similarity. Decision-making Unit 540 can anticipate any Instruction Sets 526 (not shown) correlated with Collection of Object Representations 525d4, thereby enabling autonomous Avatar 605 operation. Decision-making Unit 540 can then perform collective Similarity Comparisons 125 of Collections of Object Representations 525l1-525l5 or portions thereof from Object Processing Unit 140 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Collection of Knowledge Cells 530d. Collections of Object Representations 525d1-525d5 or portions thereof from Knowledge Cell 800rd may be found substantially similar with highest similarity. Decision-making Unit 540 can anticipate any Instruction Sets 526 (not shown) correlated with Collection of Object Representations 525d5, thereby enabling autonomous Avatar 605 operation. Decision-making Unit 540 can implement similar logic or process for any additional Collections of Object Representations 525 from Object Processing Unit 140, and so on.

In some embodiments, various elements and/or techniques can be utilized in the aforementioned similarity determinations with respect to collectively compared Collections of Object Representations 525 and/or other elements. In some aspects, similarity of collectively compared Collections of Object Representations 525 can be determined based on similarities or similarity indexes of the individually compared Collections of Object Representations 525. In one example, an average of similarities or similarity indexes of individually compared Collections of Object Representations 525 can be used to determine similarity of collectively compared Collections of Object Representations 525. In another example, a weighted average of similarities or similarity indexes of individually compared Collections of Object Representations 525 can be used to determine similarity of collectively compared Collections of Object Representations 525. For instance, to affect the weighting of collective similarity, a higher weight or importance (i.e. importance index, etc.) can be assigned to the similarities or similarity indexes of some (i.e. more substantive or larger, etc.) Collections of Object Representations 525 and lower for other (i.e. less substantive or smaller, etc.) Collections of Object Representations 525. Any other higher or lower weight or importance assignment can be implemented. In other aspects, any of the previously described or other thresholds for substantial similarity of individually compared elements can be similarly utilized for collectively compared elements. In one example, substantial similarity of collectively compared Collections of Object Representations 525

can be achieved when their collective similarity or similarity index exceeds a similarity threshold. In another example, substantial similarity of collectively compared Collections of Object Representations 525 can be achieved when at least a threshold number or percentage of Collections of Object Representations 525 or portions thereof from the collectively compared Collections of Object Representations 525 match or substantially match. Similarly, substantial similarity of collectively compared Collections of Object Representations 525 can be achieved when a number or percentage of matching or substantially matching Collections of Object Representations 525 or portions thereof from the collectively compared Collections of Object Representations 525 exceeds a threshold. Such thresholds can be defined by a user, by ACAAO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. Similar elements and/or techniques as the aforementioned can be used for similarity determinations of other collectively compared elements such as Object Representations 625, Object Properties 630, Instruction Sets 526, Extra Info 527, Knowledge Cells 800, and/or others. Similarity determinations of collectively compared elements may include any features, functionalities, and embodiments of Similarity Comparison 125, and vice versa.

It should be understood that any of the described elements and/or techniques in the foregoing example can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells 800 or elements (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info 527 in Similarity Comparisons 125, traversing of Knowledge Cells 800 or other elements, using history of Collections of Object Representations 525 or Knowledge Cells 800 for collective Similarity Comparisons 125, using various arrangements of Collections of Object Representations 525 and/or other elements in a Knowledge Cell 800, and/or others can similarly be utilized in this example. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to Fig. 26, an embodiment of determining anticipatory Instruction Sets 526 using Neural Network 530a is illustrated. In some aspects, determining anticipatory Instruction Sets 526 using Neural Network 530a may include selecting a path of Knowledge Cells 800 or elements (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof through Neural Network 530a. Decision-making Unit 540 can utilize various elements and/or techniques for selecting a path through Neural Network 530a. Although, these elements and/or techniques are described with respect to Neural Network 530a below, they can similarly be used in any Knowledgebase 530 (i.e. Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.) as applicable.

In some embodiments, Decision-making Unit 540 can utilize similarity index in selecting Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof in a path through Neural Network 530a. For instance, similarity index may indicate how well one Knowledge Cell 800 or portions thereof are matched with another Knowledge Cell 800 or portions thereof as previously described. In one example, Decision-making Unit 540 may select a Knowledge Cell 800 comprising one or more Collections of Object Representations 525 with highest similarity index even if Connection 853 pointing to that Knowledge Cell 800 has less than the highest weight. Therefore, similarity index or other such element or parameter can override or

disregard the weight of a Connection 853 or other element. In another example, Decision-making Unit 540 may select a Knowledge Cell 800 comprising one or more Collections of Object Representations 525 whose similarity index is higher than or equal to a weight of Connection 853 pointing to that Knowledge Cell 800. In a further example, Decision-making Unit 540 may select a Knowledge Cell 800 comprising one or more Collections of Object Representations 525 whose similarity index is lower than or equal to a weight of Connection 853 pointing to that Knowledge Cell 800. Similarity index can be set to be more, less, or equally important than a weight of a Connection 853.

In some embodiments, Decision-making Unit 540 can utilize Connections 853 in selecting Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof in a path through Neural Network 530a. In some aspects, Decision-making Unit 540 can take into account weights of Connections 853 among the interconnected Knowledge Cells 800 in choosing from which Knowledge Cell 800 to compare one or more Collections of Object Representations 525 first, second, third, and so on. Specifically, for instance, Decision-making Unit 540 can perform Similarity Comparisons 125 with one or more Collections of Object Representations 525 from Knowledge Cell 800 pointed to by the highest weight Connection 853 first, Collections of Object Representations 525 from Knowledge Cell 800 pointed to by the second highest weight Connection 853 second, and so on. In other aspects, Decision-making Unit 540 can stop performing Similarity Comparisons 125 as soon as it finds one or more substantially similar Collections of Object Representations 525 in an interconnected Knowledge Cell 800. In further aspects, Decision-making Unit 540 may only follow the highest weight Connection 853 to arrive at a Knowledge Cell 800 comprising one or more Collections of Object Representations 525 to be compared, thereby disregarding Connections 853 with less than the highest weight. In further aspects, Decision-making Unit 540 may ignore weights and/or other parameters of Connections 853. In further aspects, Decision-making Unit 540 may ignore Connections 853.

In some embodiments, Decision-making Unit 540 can utilize a bias to adjust similarity index, weight of a Connection 853, and/or other element or parameter used in selecting Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof in a path through Neural Network 530a. In one example, Decision-making Unit 540 may select a Knowledge Cell 800 comprising one or more Collections of Object Representations 525 whose similarity index multiplied by or adjusted for a bias is higher than or equal to a weight of Connection 853 pointing to that Knowledge Cell 800. In another example, Decision-making Unit 540 may select a Knowledge Cell 800 comprising one or more Collections of Object Representations 525 whose similarity index multiplied by or adjusted for a bias is lower than or equal to a weight of Connection 853 pointing to that Knowledge Cell 800. In a further example, bias can be used to resolve deadlock situations where similarity index is equal to a weight of a Connection 853. In some aspects, bias can be expressed in percentages such as 0.3 percent, 1.2 percent, 25.7 percent, 79.8 percent, 99.9 percent, 100.1 percent, 155.4 percent, 298.6 percent, 1105.5 percent, and so on. For example, a bias below 100 percent decreases an element or parameter to which it is applied, a bias equal to 100 percent does not change the element or parameter to which it is applied, and a bias higher than 100 percent increases the element or parameter to which it is applied. In general, any amount of bias can be utilized depending on implementation. Bias can be applied to one or more of a weight of a Connection 853, similarity index, any other element or parameter, and/or all or any combination of them. Also, different biases can be applied to each of a weight of a Connection 853, similarity index, or any other element or parameter. For

example, 30 percent bias can be applied to similarity index and 15 percent bias can be applied to a weight of a Connection 853. Also, different biases can be applied to various Layers 854 of Neural Network 530a, and/or other disclosed elements. Bias can be defined by a user, by ACAAO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input.

Any other element and/or technique can be utilized in selecting Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof in a path through Neural Network 530a.

In some embodiments, Neural Network 530a may include knowledge (i.e. interconnected Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527, etc.) of how Avatar 605 operated in circumstances including objects with various properties. In some aspects, determining anticipatory Instruction Sets 526 using Neural Network 530a may include selecting a path of Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof through Neural Network 530a. Individual and/or collective Similarity Comparisons 125 can be used to determine substantial similarity of the individually and/or collectively compared Collections of Object Representations 525 or portions thereof. Substantial similarity may be used primarily for selecting a path through Neural Network 530a, whereas, weight of any Connection 853 may be used secondarily or not at all.

For example, Decision-making Unit 540 can perform Similarity Comparisons 125 of Collections of Object Representations 525a1-525an or portions thereof from Object Processing Unit 140 with Collections of Object Representations 525 or portions thereof from one or more Knowledge Cells 800 in Layer 854a (or any other one or more Layers 854, etc.). Collections of Object Representations 525 or portions thereof from Knowledge Cell 800ta may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Avatar 605 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object Representations 525b1-525bn or portions thereof from Object Processing Unit 140 with Collections of Object Representations 525 or portions thereof from one or more Knowledge Cells 800 in Layer 854b interconnected with Knowledge Cell 800ta. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800tb may be found collectively substantially similar with highest similarity, thus, Decision-making Unit 540 may follow Connection 853t1 disregarding its less than highest weight. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Avatar 605 operation. Since Connection 853t2 is the only connection from Knowledge Cell 800tb, Decision-making Unit 540 may follow Connection 853t2 and perform Similarity Comparisons 125 of Collections of Object Representations 525c1-525cn or portions thereof from Object Processing Unit 140 with Collections of Object Representations 525 or portions thereof from Knowledge Cell 800tc in Layer 854c. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800tc may be found collectively substantially similar. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar

individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Avatar 605 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object Representations 525d1-525dn or portions thereof from Object Processing Unit 140 with Collections of Object Representations 525 or portions thereof from one or more Knowledge Cells 800 in Layer 854d interconnected with Knowledge Cell 800tc. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800td may be found collectively substantially similar with highest similarity, thus, Decision-making Unit 540 may follow Connection 853t3. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Avatar 605 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object Representations 525e1-525en or portions thereof from Object Processing Unit 140 with Collections of Object Representations 525 or portions thereof from one or more Knowledge Cells 800 in Layer 854e interconnected with Knowledge Cell 800td. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800te may be found collectively substantially similar with highest similarity, thus, Decision-making Unit 540 may follow Connection 853t4. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Avatar 605 operation. Decision-making Unit 540 can implement similar logic or process for any additional Collections of Object Representations 525 from Object Processing Unit 140.

The foregoing exemplary embodiment provides an example of utilizing a combination of collective Similarity Comparisons 125, individual Similarity Comparisons 125, Connections 853, and/or other elements or techniques. It should be understood that any of these elements and/or techniques can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof in a path through Neural Network 530a would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info 527 in Similarity Comparisons 125, traversing of Knowledge Cells 800 or other elements, using history of Collections of Object Representations 525 or Knowledge Cells 800 for collective Similarity Comparisons 125, using various arrangements of Collections of Object Representations 525 and/or other elements in a Knowledge Cell 800, and/or others can similarly be utilized in this example. These elements and/or techniques can similarly be utilized in Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, and/or other data structures or arrangements. In some aspects, instead of anticipating Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar streams of Collections of Object Representations 525. In other aspects, any time that substantial similarity or other similarity threshold is not achieved in compared Collections of Object Representations 525 or portions thereof from any of the Knowledge Cells 800, Decision-making Unit 540 can decide to look for a substantially or otherwise similar Collections of Object Representations 525 or portions thereof in Knowledge Cells 800 elsewhere in Neural Network 530a such as in any Layer 854 subsequent to a current Layer 854, in the first Layer 854, in the entire Neural Network 530a, and/or others, even if such Knowledge Cell 800 may be unconnected with a prior Knowledge

Cell 800. It should be noted that any of Collections of Object Representations 525a1-525an, 525b1-525bn, 525c1-525cn, 525d1-525dn, 525e1-525en, etc. may include one Collection of Object Representations 525 or a stream of Collections of Object Representations 525. It should also be noted that any Knowledge Cell 800 may include one Collection of Object Representations 525 or a stream of Collections of Object Representations 525 as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to Fig. 27, an embodiment of determining anticipatory Instruction Sets 526 using Graph 530b is illustrated. Graph 530b may include knowledge (i.e. interconnected Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527, etc.) of how Avatar 605 operated in circumstances including objects with various properties. In some aspects, determining anticipatory Instruction Sets 526 using Graph 530b may include selecting a path of Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof through Graph 530b. Individual and/or collective Similarity Comparisons 125 can be used to determine substantial similarity of the individually and/or collectively compared Collections of Object Representations 525 or portions thereof. Substantial similarity may be used primarily for selecting a path through Graph 530b, whereas, weight of any Connection 853 may be used secondarily or not at all.

For example, Decision-making Unit 540 can perform Similarity Comparisons 125 of Collections of Object Representations 525a1-525an or portions thereof from Object Processing Unit 140 with Collections of Object Representations 525 or portions thereof from one or more Knowledge Cells 800 in Graph 530b. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800ua may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Avatar 605 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object Representations 525b1-525bn or portions thereof from Object Processing Unit 140 with Collections of Object Representations 525 or portions thereof from one or more Knowledge Cells 800 in Graph 530b interconnected with Knowledge Cell 800ua by outgoing Connections 853. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800ub may be found collectively substantially similar with highest similarity, thus, Decision-making Unit 540 may follow Connection 853u1 disregarding its less than highest weight. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Avatar 605 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object Representations 525c1-525cn or portions thereof from Object Processing Unit 140 with Collections of Object Representations 525 or portions thereof from one or more Knowledge Cells 800 in Graph 530b interconnected with Knowledge Cell 800ub by outgoing Connections 853. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800uc may be found collectively substantially similar with highest similarity, thus, Decision-making Unit 540 may follow Connection 853u2 disregarding its less than highest weight. As the comparisons of

individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Avatar 605 operation. Since Connection 853u3 is the only connection from Knowledge Cell 800uc, Decision-making Unit 540 may follow Connection 853u3 and perform Similarity Comparisons 125 of Collections of Object Representations 525d1-525dn or portions thereof from Object Processing Unit 140 with Collections of Object Representations 525 or portions thereof from Knowledge Cell 800ud in Graph 530b. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800ud may be found collectively substantially similar. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Avatar 605 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object Representations 525e1-525en or portions thereof from Object Processing Unit 140 with Collections of Object Representations 525 or portions thereof from one or more Knowledge Cells 800 in Graph 530b interconnected with Knowledge Cell 800ud by outgoing Connections 853. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800ue may be found collectively substantially similar with highest similarity, thus, Decision-making Unit 540 may follow Connection 853u4. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Avatar 605 operation. Decision-making Unit 540 can implement similar logic or process for any additional Collections of Object Representations 525 from Object Processing Unit 140.

The foregoing exemplary embodiment provides an example of utilizing a combination of collective Similarity Comparisons 125, individual Similarity Comparisons 125, Connections 853, and/or other elements or techniques. It should be understood that any of these elements and/or techniques can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof in a path through Graph 530b would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info 527 in Similarity Comparisons 125, traversing of Knowledge Cells 800 or other elements, using history of Collections of Object Representations 525 or Knowledge Cells 800 in collective Similarity Comparisons 125, using various arrangements of Collections of Object Representations 525 and/or other elements in a Knowledge Cell 800, and/or others can similarly be utilized in this example. These elements and/or techniques can similarly be utilized in Neural Network 530a, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, and/or other data structures or arrangements. In some aspects, instead of anticipating Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially matching streams of Collections of Object Representations 525. In other aspects, any time that substantial similarity or other similarity threshold is not achieved in compared Collections of Object Representations 525 or portions thereof of any of the Knowledge Cells 800, Decision-making Unit 540 can decide to look for a substantially or otherwise similar Collections of Object Representations 525 or portions thereof in Knowledge Cells 800 elsewhere in

Graph 530b even if such Knowledge Cell 800 may be unconnected with a prior Knowledge Cell 800. It should be noted that any of Collections of Object Representations 525a1-525an, 525b1-525bn, 525c1-525cn, 525d1-525dn, 525e1-525en, etc. may include one Collection of Object Representations 525 or a stream of Collections of Object Representations 525. It should also be noted that any Knowledge Cell 800 may include one Collection of Object Representations 525 or a stream of Collections of Object Representations 525 as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to Fig. 28, an embodiment of determining anticipatory Instruction Sets 526 using Collection of Sequences 530c is illustrated. Collection of Sequences 530c may include knowledge (i.e. sequences of Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527, etc.) of how Avatar 605 operated in circumstances including objects with various properties. In some aspects, determining anticipatory Instruction Sets 526 for autonomous Avatar 605 operation using Collection of Sequences 530c may include selecting a Sequence 533 of Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof from Collection of Sequences 530c. Individual and/or collective Similarity Comparisons 125 can be used to determine substantial similarity of the individually and/or collectively compared Collections of Object Representations 525 or portions thereof.

For example, Decision-making Unit 540 can perform Similarity Comparisons 125 of Collections of Object Representations 525a1-525an or portions thereof from Object Processing Unit 140 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in one or more Sequences 533 of Collection of Sequences 530c. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800ca in Sequence 533wc may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Avatar 605 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object Representations 525a1-525an and 525b1-525bn or portions thereof from Object Processing Unit 140 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Sequences 533 of Collection of Sequences 530c. Collections of Object Representations 525 or portions thereof from Knowledge Cells 800ca-800cb in Sequence 533wc may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Avatar 605 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object Representations 525a1-525an, 525b1-525bn, and 525c1-525cn or portions thereof from Object Processing Unit 140 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Sequences 533 of Collection of Sequences 530c. Collections of Object Representations 525 or portions thereof from Knowledge Cells 800da-800dc in Sequence 533wd may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-

making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Avatar 605 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object Representations 525a1-525an, 525b1-525bn, 525c1-525cn, and 525d1-525dn or portions thereof from Object Processing Unit 140 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Sequences 533 of Collection of Sequences 530c. Collections of Object Representations 525 or portions thereof from Knowledge Cells 800da-800dd in Sequence 533wd may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Avatar 605 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object Representations 525a1-525an, 525b1-525bn, 525c1-525cn, 525d1-525dn, and 525e1-525en or portions thereof from Object Processing Unit 140 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Sequences 533 of Collection of Sequences 530c. Collections of Object Representations 525 or portions thereof from Knowledge Cells 800da-800de in Sequence 533wd may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Avatar 605 operation. Decision-making Unit 540 can implement similar logic or process for any additional Collections of Object Representations 525 from Object Processing Unit 140, and so on.

The foregoing exemplary embodiment provides an example of utilizing a combination of collective Similarity Comparisons 125, individual Similarity Comparisons 125, and/or other elements or techniques. It should be understood that any of these elements and/or techniques can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Sequence 533 of Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info 527 in Similarity Comparisons 125, traversing of Knowledge Cells 800 or other elements, using history of Collections of Object Representations 525 or Knowledge Cells 800 in collective Similarity Comparisons 125, using various arrangements of Collections of Object Representations 525 and/or other elements in a Knowledge Cell 800, and/or others can similarly be utilized in this example. These elements and/or techniques can similarly be utilized in Neural Network 530a, Graph 530b, Collection of Knowledge Cells 530d, and/or other data structures or arrangements. In some aspects, instead of anticipating Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially matching streams of Collections of Object Representations 525. In other aspects, any time that substantial similarity or other similarity threshold is not achieved in compared Collections of Object Representations 525 or portions thereof from any of the Knowledge Cells 800, Decision-making Unit 540 can decide to look for a substantially or otherwise similar Collections of Object Representations 525 or portions thereof in Knowledge Cells 800 elsewhere in Collection of Sequences 530c such as in different Sequences 533. It should be noted that any of Collections of Object Representations 525a1-525an, 525b1-525bn,

525c1-525cn, 525d1-525dn, 525e1-525en, etc. may include one Collection of Object Representations 525 or a stream of Collections of Object Representations 525. It should also be noted that any Knowledge Cell 800 may include one Collection of Object Representations 525 or a stream of Collections of Object Representations 525 as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring now to Modification Interface 130. Modification Interface 130 comprises the functionality for modifying execution and/or functionality of Avatar 605, Application Program 18, Processor 11, and/or other processing element. Modification Interface 130 comprises the functionality for modifying execution and/or functionality of Avatar 605, Application Program 18, Processor 11, and/or other processing element at runtime. Modification Interface 130 comprises the functionality for modifying execution and/or functionality of Avatar 605, Application Program 18, Processor 11, and/or other processing element based on anticipatory Instruction Sets 526. In one example, Artificial Intelligence Unit 110 may determine anticipatory Instruction Sets 526 to be used or executed in Avatar's 605 autonomous operation, and Modification Interface 130 may use the anticipatory Instruction Sets 526 to modify Avatar 605 to effect Avatar's 605 autonomous operation. In another example, in Application Programs 18 that do not comprise Avatar 605 or do not rely on Avatar 605 for their operation, Artificial Intelligence Unit 110 may determine anticipatory Instruction Sets 526 to be used or executed in Application Program's 18 autonomous operation, and Modification Interface 130 may use the anticipatory Instruction Sets 526 to modify Application Program 18 to effect Application Program's 18 autonomous operation. In some aspects, Modification Interface 130 can access, modify, and/or otherwise manipulate runtime engine/environment, virtual machine, operating system, compiler, interpreter, translator, execution stack, file, object, data structure, and/or other computing system elements. In other aspects, Modification Interface 130 can access, modify, and/or otherwise manipulate memory, storage, and/or other repositories. In further aspects, Modification Interface 130 can access, modify, and/or otherwise manipulate Processor 11 registers and/or other Processor 11 elements. In further aspects, Modification Interface 130 can access, modify, and/or otherwise manipulate inputs and/or outputs of Avatar 605, Application Program 18, Processor 11, and/or other processing element. In further aspects, Modification Interface 130 can access, create, delete, modify, and/or otherwise manipulate functions, methods, procedures, routines, subroutines, and/or other elements of Avatar 605 and/or Application Program 18. In further aspects, Modification Interface 130 can access, create, delete, modify, and/or otherwise manipulate source code, bytecode, compiled code, interpreted code, translated code, machine code, and/or other code. In further aspects, Modification Interface 130 can access, create, delete, modify, and/or otherwise manipulate values, variables, parameters, and/or other data or information. Modification Interface 130 comprises any features, functionalities, and embodiments of Acquisition Interface 120, and vice versa, as applicable. Modification Interface 130 also comprises other disclosed functionalities.

Modification Interface 130 can employ various techniques for modifying execution and/or functionality of Avatar 605, Application Program 18, Processor 11, and/or other processing element. In some aspects, some of the previously described techniques and/or tools can be utilized. Code instrumentation, for instance, may involve inserting additional code, overwriting or rewriting existing code, and/or branching to a separate segment of code from Application Program 18 as previously described. For example, instrumented code may include the following:

```
Statement1;
Statement2;
modifyAvatar();
Statement3;
Statement4;
```

In the above sample code, instrumented call to Modification Interface's 130 function such as modifyAvatar(), modifyApplication(), etc. can be placed before or after any statement of Avatar 605 and/or Application Program 18 such as after Statement2. A similar call to Modification Interface's 130 function that modifies Avatar 605 and/or Application Program 18 can be placed before or after some or all functions/routines/subroutines, some or all lines of code, some or all statements, some or all instructions or instruction sets, some or all basic blocks, and/or some or all other code segments of Avatar 605 and/or Application Program 18. One or more calls to functions that modify Avatar 605 and/or Application Program 18 can be placed anywhere in Avatar's 605 and/or Application Program's 18 code and can be executed at any points in Avatar's 605 and/or Application Program's 18 execution. A function that modifies Avatar 605 and/or Application Program 18 may include Artificial Intelligence Unit 110-determined anticipatory Instruction Sets 526 to be used or executed in Avatar's 605 and/or Application Program's 18 autonomous operation. In some embodiments, the previously described obtaining Avatar's 605 and/or Application Program's 18 instruction sets, data, and/or other information as well as modifying execution and/or functionality of Avatar 605 and/or Application Program 18 can be implemented in a single function that performs both tasks such as traceAndModifyAvatar(), traceAndModifyApplication(), etc.

In some embodiments, various computing systems and/or platforms may provide native tools for modifying execution and/or functionality of Avatar 605, Application Program 18, Processor 11, and/or other processing element. Independent vendors may provide tools with similar functionalities that can be utilized across different platforms. These tools enable a wide range of techniques or capabilities such as instrumentation, self-modifying code capabilities, dynamic code capabilities, branching, code rewriting, code overwriting, hot swapping, accessing and/or modifying objects or data structures, accessing and/or modifying functions/routines/subroutines, accessing and/or modifying variable or parameter values, accessing and/or modifying processor registers, accessing and/or modifying inputs and/or outputs, providing runtime memory access, and/or other capabilities. One of ordinary skill in art will understand that, while all possible variations of the techniques for modifying execution and/or functionality of Avatar 605, Application Program 18, Processor 11, and/or other processing element are too voluminous to describe, these techniques are within the scope of this disclosure.

In one example, modifying execution and/or functionality of Avatar 605 and/or Application Program 18 can be implemented through utilizing metaprogramming techniques, which include applications that can self-modify or that can create, modify, and/or manipulate other applications. Self-modifying code, dynamic code, reflection, and/or other techniques can be used to facilitate metaprogramming. In some aspects, metaprogramming is facilitated through a programming language's ability to access and manipulate the internals of the runtime engine directly or via an API. In other aspects, metaprogramming is facilitated through dynamic execution of expressions (i.e. anticipatory Instruction Sets 526, etc.) that can be created and/or executed at runtime. In yet other aspects, metaprogramming is facilitated through application modification tools, which can perform modifications on an application regardless of whether the application's programming language enables metaprogramming capabilities. Some operating systems

may protect an application loaded into memory by restricting access to the loaded application. This protection mechanism can be circumvented by utilizing operating system's, processor's, and/or other low level features or commands to unprotect the loaded application. For example, a self-modifying application may modify the in-memory image of itself. To do so, the application can obtain the in-memory address of its code. The application may then change the operating system's or platform's protection on this memory range allowing it to modify the code (i.e. insert anticipatory Instruction Sets 526, etc.). In addition to a self-modifying application, one application can utilize similar technique to modify another application. Linux mprotect command or similar commands of other operating systems can be used to change protection (i.e. unprotect, etc.) for a region of memory, for example. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Avatar 605 and/or Application Program 18 can be implemented through native capabilities of dynamic, interpreted, and/or scripting programming languages and/or platforms. Most of these languages and/or platforms can perform functionalities at runtime that static programming languages may perform during compilation. Dynamic, interpreted, and/or scripting languages provide native functionalities such as self-modification of code, dynamic code, extending the application, adding new code, extending objects and definitions, and/or other functionalities that can modify an application's execution and/or functionality at runtime. Examples of dynamic, interpreted, and/or scripting languages include Lisp, Perl, PHP, JavaScript, Ruby, Python, Smalltalk, Tcl, VBScript, and/or others. Similar functionalities can also be provided in languages such as Java, C, and/or others using reflection. Reflection includes the ability of an application to examine and modify the structure and behavior of the application at runtime. For example, JavaScript can modify its own code as it runs by utilizing Function object constructor as follows:

```
myFunc=new Function(arg1, arg2, argN, functionBody);
```

The sample code above causes a new function object to be created with the specified arguments and body. The body and/or arguments of the new function object may include new instruction sets (i.e. anticipatory Instruction Sets 526, etc.). The new function can be invoked as any other function in the original code. In another example, JavaScript can utilize eval method that accepts a string of JavaScript statements (i.e. anticipatory Instruction Sets 526, etc.) and execute them as if they were within the original code. An example of how eval method can be used to modify an application includes the following JavaScript code:

```
anticipatoryInstr = ‘Avatar.moveForward(14);’;
if (anticipatoryInstr != "" && anticipatoryInstr != null)
{
    eval(anticipatoryInstr);
}
```

In the sample code above, Artificial Intelligence Unit 110 may generate anticipatory Instruction Set 526 (i.e. ‘Avatar.moveForward(14)’ for moving Avatar 605 forward 14 units, etc.) and save it in anticipatoryInstr variable, which eval method can then execute. Lisp is another example of dynamic, interpreted, and/or scripting language that includes similar capabilities as the aforementioned JavaScript. For example, Lisp's compile command can create a function at runtime, eval command may parse and evaluate an expression at runtime, and exec command may execute a given instruction set (i.e. string, etc.) at runtime. In another example, dynamic as well as some non-dynamic languages may provide macros, which combine code introspection and/or eval capabilities. In some

aspects, macros can access inner workings of the compiler, interpreter, virtual machine, runtime environment/engine, and/or other components of the computing platform enabling the definition of language-like constructs and/or generation of a complete program or sections thereof. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Avatar 605 and/or Application Program 18 can be implemented through dynamic code, dynamic class loading, reflection, and/or other functionalities of a programming language or platform. In static applications or static programming, a class can be defined and/or loaded at compile time. Conversely, in dynamic applications or dynamic programming, a class can be loaded into a running environment at runtime. For example, Java Runtime Environment (JRE) may not require that all classes be loaded at compile time and class loading can occur when a class is first referenced at runtime. Dynamic class loading enables inclusion or injection of on-demand code and/or functionalities at runtime. System provided or custom class loaders may enable loading of classes into the running environment. Custom class loaders can be created to enable custom functionalities such as, for example, specifying a remote location from which a class can be loaded. In addition to dynamic loading of a pre-defined class, a class can also be created at runtime. In some aspects, a class source code can be created at runtime. A compiler such as javac, com.sun.tools.javac.Main, javax.tools, javax.tools.JavaCompiler, and/or other packages can then be utilized to compile the source code. Javac, com.sun.tools.javac.Main, javax.tools, javax.tools.JavaCompiler, and/or other packages may include an interface to invoke Java compiler from within a running application. A Java compiler may accept source code in a file, string, object (i.e. Java String, StringBuffer, CharSequence, etc.) and/or other source, and may generate Java bytecode (i.e. class file, etc.). Once compiled, a class loader can then load the compiled class into the running environment. In other aspects, a tool such as Javaassist (i.e. Java programming assistant) can be utilized to enable an application to create or modify a class at runtime. Javassist may include a Java library that provides functionalities to create and/or manipulate Java bytecode of an application as well as reflection capabilities. Javassist may provide source-level and bytecode-level APIs. Using the source-level API, a class can be created and/or modified using only source code, which Javassist may compile seamlessly on the fly. Javassist source-level API can therefore be used without knowledge of Java bytecode specification. Bytecode-level API enables creating and/or editing a class bytecode directly. In yet other aspects, similar functionalities to the aforementioned ones may be provided in tools such as Apache Commons BCEL (Byte Code Engineering Library), ObjectWeb ASM, CGLIB (Byte Code Generation Library), and/or others. Once a dynamic code or class is created and loaded, reflection in high-level programming languages such as Java and/or others can be used to manipulate or change the runtime behavior of an application. Examples of reflective programming languages and/or platforms include Java, JavaScript, Smalltalk, Lisp, Python, .NET Common Language Runtime (CLR), Tcl, Ruby, Perl, PHP, Scheme, PL/SQL, and/or others. Reflection can be used in an application to access, examine, modify, and/or manipulate a loaded class and/or its elements. Reflection in Java can be implemented by utilizing a reflection API such as java.lang.Reflect package. The reflection API provides functionalities such as, for example, loading or reloading a class, instantiating a new instance of a class, determining class and instance methods, invoking class and instance methods, accessing and manipulating a class, fields, methods and constructors, determining the modifiers for fields, methods, classes, and interfaces, and/or other functionalities. The above described dynamic code, dynamic class loading, reflection, and/or other functionalities are similarly provided in the .NET platform through its tools such as, for example, System.CodeDom.Compiler

namespace, System.Reflection.Emit namespace, and/or other native or other .NET tools. Other platforms in addition to Java and .NET may provide similar tools and/or functionalities. In some designs, dynamic code, dynamic class loading, reflection, and/or other functionalities can be used to facilitate modification of an application by inserting or injecting instruction sets (i.e. anticipatory Instruction Sets 526, etc.) into a running application. For example, an existing or dynamically created class comprising ACAAO Unit 100 functionalities can be loaded into a running application through manual, automatic, or dynamic instrumentation. Once the class is created and loaded, an instance of ACAAO Unit 100 class may be constructed. The instance of ACAAO Unit 100 can then take or exert control of the application and/or implement alternate instruction sets (i.e. anticipatory Instruction Sets 526, etc.) at any point in the application's execution. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Avatar 605 and/or Application Program 18 can be implemented through independent tools that can be utilized across different platforms. Such tools provide instrumentation and/or other capabilities on more than one platform or computing system and may facilitate application modification or insertion of instruction sets (i.e. anticipatory Instruction Sets 526, etc.). Examples of these tools include Pin, DynamoRIO, DynInst, Kprobes, KernInst, OpenPAT, DTrace, SystemTap, and/or others. In some aspects, Pin and/or any of its elements, methods, and/or techniques can be utilized for dynamic instrumentation. Pin can perform instrumentation by taking control of an application after it loads into memory. Pin may insert itself into the address space of an executing application enabling it to take control. Pin JIT compiler can then compile and implement alternate code (i.e. anticipatory Instruction Sets 526, etc.). Pin provides an extensive API for instrumentation at several abstraction levels. Pin supports two modes of instrumentation, JIT mode and probe mode. JIT mode uses a just-in-time compiler to insert instrumentation and recompile program code while probe mode uses code trampolines for instrumentation. Pin was designed for architecture and operating system independence. In other aspects, KernInst and/or any of its elements, methods, and/or techniques can be utilized for dynamic instrumentation. KernInst includes an instrumentation framework designed for dynamically inserting code into a running kernel of an operating system. KernInst implements probe-based dynamic instrumentation where code can be inserted, changed, and/or removed at will. KernInst API enables client tools to construct their own tools for dynamic kernel instrumentation to suit variety of purposes such as insertion of alternate instruction sets (i.e. anticipatory Instruction Sets 526, etc.). Client tools can communicate with KernInst over a network (i.e. internet, wireless network, LAW, WAN, etc). Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Avatar 605 and/or Application Program 18 can be implemented through utilizing operating system's native tools or capabilities such as Unix ptrace command. Ptrace includes a system call that may enable one process to control another allowing the controller to inspect and manipulate the internal state of its target. Ptrace can be used to modify a running application such as modify an application with alternate instruction sets (i.e. anticipatory Instruction Sets 526, etc.). By attaching to an application using the ptrace call, the controlling application can gain extensive control over the operation of its target. This may include manipulation of its instruction sets, execution path, file descriptors, memory, registers, and/or other components. Ptrace can single-step through the target's code, observe and intercept system calls and their results, manipulate the target's signal handlers, receive and send signals on the target's behalf, and/or perform other

operations within the target application. Ptrace's ability to write into the target application's memory space enables the controller to modify the running code of the target application. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Avatar 605 and/or Application Program 18 can be implemented through utilizing just-in-time (JIT) compiling. JIT compilation (also known as dynamic translation, dynamic compilation, etc.) includes compilation performed during an application's execution (i.e. runtime, etc.). A code can be compiled when it is about to be executed, and it may be cached and reused later without the need for additional compilation. In some aspects, a JIT compiler can convert source code or byte code into machine code. In other aspects, a JIT compiler can convert source code into byte code. JIT compiling may be performed directly in memory. For example, JIT compiler can output machine code directly into memory and immediately execute it. Platforms such as Java, .NET, and/or others may implement JIT compilation as their native functionality. Platform independent tools for custom system design may include JIT compilation functionalities as well. In some aspects, JIT compilation includes redirecting application's execution to a JIT compiler from a specific entry point. For example, Pin can insert its JIT compiler into the address space of an application. Once execution is redirected to it, JIT compiler may receive alternate instruction sets (i.e. anticipatory Instruction Sets 526, etc.) immediately before their compilation. The JIT compiled instruction sets may be stored in memory or another repository from where they can be retrieved and executed. Alternatively, for example, JIT compiler can create a copy of the original application code or a segment thereof and insert alternate code (i.e. anticipatory Instruction Sets 526, etc.) before compiling the modified code copy. In some aspects, JIT compiler may include a specialized memory such as fast cache memory dedicated to JIT compiler functionalities from which the modified code can be fetched rapidly. JIT compilation and/or any compilation in general may include compilation, interpretation, or other translation into machine code, bytecode, and/or other formats or types of code. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Avatar 605 and/or Application Program 18 can be implemented through dynamic recompilation. Dynamic recompilation includes recompiling an application or part thereof during execution. An application can be modified with alternate features or instruction sets that may take effect after recompilation. Dynamic recompilation may be practical in various types of applications including object oriented, event driven, forms based, and/or other applications. In a typical windows-based application, most of the action after initial startup occurs in response to user or system events such as moving the mouse, selecting a menu option, typing text, running a scheduled task, making a network connection, and/or other events when an event handler is called to perform an operation appropriate for the event. Generally, when no events are being generated, the application is idle. For example, when an event occurs and an appropriate event handler is called, instrumentation can be implemented in the application's source code to insert alternate instruction sets (i.e. anticipatory Instruction Sets 526, etc.) at which point the modified source code can be recompiled and/or executed. In some aspects, the state of the application can be saved before recompiling its modified source code so that the application may continue from its prior state. Saving the application's state can be achieved by saving its variables, data structures, objects, location of its current instruction, and/or other necessary information in environmental variables, memory, or other repositories where they can be accessed once the application is recompiled. In other aspects, application's variables, data structures, objects, address of its current instruction, and/or other necessary

information can be saved in a repository such as file, database, or other repository accessible to the application after recompilation of its source code. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Avatar 605 and/or Application Program 18 can be implemented through modifying or redirecting application's execution path. Generally, an application can be loaded into memory and the flow of execution proceeds from one instruction set to the next until the end of the application. An application may include a branching mechanism that can be driven by keyboard or other input devices, system events, and/or other computing system components or events that may impact the execution path. The execution path can also be altered by an external application through acquiring control of execution and/or redirecting execution to a function, routine/subroutine, or an alternate code segment at any point in the application's execution. A branch, jump, or other mechanism can be utilized to implement the redirected execution. For example, a jump instruction can be inserted at a specific point in an application's execution to redirect execution to an alternate code segment. A jump instruction set may include, for example, an unconditional branch, which always results in branching, or a conditional branch, which may or may not result in branching depending on a condition. When executing an application, a computer may fetch and execute instruction sets in sequence until it encounters a branch instruction set. If the instruction set is an unconditional branch, or it is conditional and the condition is satisfied, the computer may fetch its next instruction set from a different instruction set sequence or code segment as specified by the branch instruction set. After the execution of the alternate code segment, control may be redirected back to the original jump point or to another point in the application. For example, modifying an application can be implemented by redirecting execution of an application to alternate instruction sets (i.e. anticipatory Instruction Sets 526, etc.). Alternate instruction sets can be pre-compiled, pre-interpreted, or otherwise pre-translated and ready for execution. Alternate instruction sets can also be JIT compiled, JIT interpreted, or otherwise JIT translated before execution. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Avatar 605 and/or Application Program 18 can be implemented through assembly language. Assembly language instructions may be directly related with the architecture's machine instructions as previously described. Assembly language can, therefore, be a powerful tool for implementing direct hardware (i.e. processor registers, memory, etc.) access and manipulations as well as access and manipulations of specialized processor features or instructions. Assembly language can also be a powerful tool for implementing low-level embedded systems, real-time systems, interrupt handlers, self or dynamically modifying code, and/or other applications. Specifically, for instance, self or dynamically modifying code that can be used to facilitate modifying of an application can be seamlessly implemented using assembly language. For example, using assembly language, instruction sets can be dynamically created and loaded into memory similar to the ones that a compiler may generate. Furthermore, using assembly language, memory space of a loaded application can be accessed to modify (including rewrite, overwrite, etc.) original instruction sets or to insert jumps or branches to alternate code elsewhere in memory. Some operating systems may implement protection from changes to applications loaded into memory. Operating system's, processor's, or other low level features or commands can be used to unprotect the protected locations in memory before the change as previously described. Alternatively, a pointer that may reside in a memory location where it could be readily altered can be utilized where the pointer may

reference alternate code. In one example, assembly language can be utilized to write alternate code (i.e. anticipatory Instruction Sets 526, etc.) into a location in memory outside a running application's memory space. Assembly language can then be utilized to redirect the application's execution to the alternate code by inserting a jump or branch into the application's in-memory code, by redirecting program counter, or by other technique. In another example, assembly language can be utilized to overwrite or rewrite the entire or part of an application's in-memory code with alternate code. In some aspects, high-level programming languages can call an external assembly language program to facilitate application modification as previously described. In yet other aspects, relatively low-level programming languages such as C may allow embedding assembly language directly in their source code such as, for example, using asm keyword of C. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Avatar 605 and/or Application Program 18 can be implemented through binary rewriting. Binary rewriting tools and/or techniques may modify an application's executable. In some aspects, modification can be minor such as in the case of optimization where the original executable's functionality is kept. In other aspects, modification may change the application's functionality such as by inserting alternate code (i.e. anticipatory Instruction Sets 526, etc.). Examples of binary rewriting tools include SecondWrite, ATOM, DynamoRIO, Purify, Pin, EEL, DynInst, PLTO, and/or others. Binary rewriting may include disassembly, analysis, and/or modification of target application. Since binary rewriting works directly on machine code executable, it is independent of source language, compiler, virtual machine (if one is utilized), and/or other higher level abstraction layers. Also, binary rewriting tools can perform application modifications without access to original source code. Binary rewriting tools include static rewriters, dynamic rewriters, minimally-invasive rewriters, and/or others. Static binary rewriters can modify an executable when the executable is not in use (i.e. not running). The rewritten executable may then be executed including any new or modified functionality. Dynamic binary rewriters can modify an executable during its execution, thereby enabling modification of an application's functionality at runtime. In some aspects, dynamic rewriters can be used for instrumentation or selective modifications such as insertion of alternate code (i.e. anticipatory Instruction Sets 526, etc.), and/or for other runtime transformations or modifications. For example, some dynamic rewriters can be configured to intercept an application's execution at indirect control transfers and insert instrumentation or other application modifying code. Minimally-invasive rewriters may keep the original machine code to the greatest extent possible. They support limited modifications such as insertion of jumps into and out of instrumented code. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

Referring to Fig. 29, in a further example, modifying execution and/or functionality of Avatar 605 and/or Application Program 18 can be implemented through modification of processor registers, memory, or other computing system components. In some aspects, modifying execution and/or functionality of Processor 11 can be implemented by redirecting Processor's 11 execution to alternate instruction sets (i.e. anticipatory Instruction Sets 526, etc.). In one example, Program Counter 211 may hold or point to a memory address of the next instruction set that will be executed by Processor 11. Artificial Intelligence Unit 110 may generate anticipatory Instruction Sets 526 and store them in Memory 12 as previously described. Modification Interface 130 may then change Program Counter 211 to point to the location in Memory 12 where anticipatory Instruction Sets 526 are stored. The anticipatory Instruction Sets 526 can then be fetched from the location in Memory 12 pointed to by the modified

Program Counter 211 and loaded into Instruction Register 212 for decoding and execution. Once anticipatory Instruction Sets 526 are executed, Modification Interface 130 may change Program Counter 211 to point to the last instruction set before the redirection or to any other instruction set. In other aspects, anticipatory Instruction Sets 526 can be loaded directly into Instruction Register 212. As previously described, examples of other processor or computing system components that can be used during an instruction cycle include memory address register (MAR), memory data register (MDR), data registers, address registers, general purpose registers (GPRs), conditional registers, floating point registers (FPRs), constant registers, special purpose registers, machine-specific registers, Register Array 214, Arithmetic Logic Unit 215, control unit, and/or other circuits or components. Any of the aforementioned processor registers, memory, or other computing system components can be accessed and/or modified to facilitate the disclosed functionalities. In some embodiments, processor interrupt may be issued to facilitate such access and/or modification. In some designs, modifying execution and/or functionality of Processor 11 can be implemented in a program, combination of programs and hardware, or purely hardware system. Dedicated hardware may be built to perform modifying execution and/or functionality of Processor 11 with marginal or no impact to computing overhead. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

Other additional techniques or elements can be utilized as needed for modifying execution and/or functionality of Avatar 605, Application Program 18, Processor 11, and/or other processing elements, or some of the disclosed techniques or elements can be excluded, or a combination thereof can be utilized in alternate embodiments. As an avatar (i.e. Avatar 605, etc.) may be part of an application (i.e. Application Program 18, etc.), it should be noted that modifying execution and/or functionality of an avatar may include same or similar techniques as the aforementioned modifying execution and/or functionality of an application, and vice versa.

Referring to Fig. 30, the illustration shows an embodiment of a method 9100 for learning and/or using an avatar's circumstances for autonomous avatar operation. In some aspects, the method can be used on a computing device or system to enable learning of an avatar's operation in circumstances including objects with various properties and enable autonomous avatar operation in similar circumstances. Method 9100 may include any action or operation of any of the disclosed methods such as method 9200, 9300, 9400, 9500, 9600, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method 9100.

At step 9105, a first collection of object representations is received. A collection of object representations (i.e. Collection of Object Representations 525, etc.) may include one or more object representations (i.e. Object Representations 625, etc.), object properties (i.e. Object Properties 630, etc.), and/or other elements or information. In some designs, an object representation includes a representation of an object (i.e. Object 615, etc.) in an avatar's (i.e. Avatar's 605, etc.) surrounding within an application (i.e. Application Program 18, etc.). In other designs, an object representation includes a representation of an object of an application. As such, an object representation may include any information related to an object. In further designs, an object representation may include or be replaced with an object itself, in which case the object representation as an element can be optionally omitted. In some embodiments, a collection of object representations may include one or more object representations, object properties, and/or other elements or information obtained in an avatar's surrounding at a particular time. A collection of object representations may, therefore, include knowledge (i.e. unit of knowledge, etc.) of an avatar's

circumstances including objects with various properties at a particular time. In other embodiments, where an application does not comprise an avatar or rely on an avatar for its operation, a collection of object representations may include one or more object representations, object properties, and/or other elements or information obtained in an application or part thereof at a particular time. A collection of object representations may, therefore, include knowledge (i.e. unit of knowledge, etc.) of an application's circumstances including objects with various properties at a particular time. In some designs, a collection of object representations may include or be associated with a time stamp (not shown), order (not shown), or other time related information. In further embodiments, a collection of object representations may include or be substituted with a stream of collections of object representations, and vice versa. Therefore, the terms collection of object representations and stream of collections of object representations may be used interchangeably depending on context. A stream of collections of object representations may include one collection of object representations or a group, sequence, or other plurality of collections of object representations. In some aspects, a stream of collections of object representations may include one or more collections of object representations, and/or other elements or information obtained in an avatar's surrounding over time. A stream of collections of object representations may, therefore, include knowledge (i.e. unit of knowledge, etc.) of an avatar's circumstances including objects with various properties over time. As circumstances including objects with various properties in an avatar's surrounding change (i.e. objects and/or their properties change, move, act, transform, etc.) over time, this change may be captured in a stream of collections of object representations. In further embodiments, where an application does not comprise an avatar or rely on an avatar for its operation, a stream of collections of object representations may include one or more collections of object representations, and/or other elements or information obtained in an application or part thereof over time. A stream of collections of object representations may, therefore, include knowledge (i.e. unit of knowledge, etc.) of an application's circumstances including objects with various properties over time. In some designs, each collection of object representations in a stream may include or be associated with the aforementioned time stamp, order, or other time related information. Examples of objects include models of a person, animal, tree, rock, building, vehicle, and/or others in a context of a computer game, virtual world, 3D or 2D graphics application, and/or others. More generally, examples of objects include a 2D model, a 3D model, a 2D shape (i.e. point, line, square, rectangle, circle, triangle, etc.), a 3D shape (i.e. cube, sphere, etc.), a graphical user interface (GUI) element, a form element (i.e. text field, radio button, push button, check box, etc.), a data or database element, a spreadsheet element, a link, a picture, a text (i.e. character, word, etc.), a number, and/or others in a context of a web browser, a media application, a word processing application, a spreadsheet application, a database application, a forms-based application, an operating system, a device/system control application, and/or others. In some aspects, any part of an object can be identified as an object itself. For instance, instead of or in addition to identifying a building as an object, a window, door, roof, and/or other parts of the building can be identified as objects. In general, an object may include any object or part thereof that can be obtained or recognized. Examples of object properties include existence of an object, type of an object (i.e. person, animal, tree, rock, building, vehicle, etc.), identity of an object (i.e. name, identifier, etc.), distance of an object, bearing/angle of an object, location of an object (i.e. distance and bearing/angle from a known point, coordinates, etc.), shape/size of an object (i.e. scale, height, width, depth, computer model, etc.), activity of an object (i.e. motion, gestures, etc.), and/or other properties of an object. In general, an object property may include any attribute of an object (i.e. existence of an object, type of an object, identity of an object, shape/size of an object,

etc.), any relationship of an object with an avatar, other objects, or the environment (i.e. distance of an object, bearing/angle of an object, friend/foe relationship, etc.), and/or other information related to an object. In some designs, objects and/or their properties can be obtained from an engine, environment, or other system used to implement an application. For instance, objects and/or their properties can be obtained by utilizing functions for providing properties or other information on objects of an engine, environment, or other system used to implement an application. Examples of such engines, environments, or other systems include Unity 3D Engine, Unreal Engine, Torque 3D Engine, and/or others. In other designs, objects and/or their properties can be obtained by accessing and/or reading a scene graph or other data structure used for organizing objects in a particular application, or in an engine, environment, or other system used to implement an application. In other designs, objects and/or their properties can be detected or recognized from one or more pictures depicting views of an avatar's surrounding or views of an application. Any picture recognition techniques (i.e. Picture Recognizer 92, etc.) can be used for such detection or recognizing. The one or more pictures depicting views of an avatar's surrounding or views of an application can be rendered or generated by a picture renderer (i.e. Picture Renderer 91, etc.). In further designs, objects and/or their properties can be detected or recognized from one or more sounds from an avatar's surrounding or one or more sounds of an application. Any sound recognition techniques (i.e. Sound Recognizer 97, etc.) can be used for such detection or recognizing. The one or more sounds from an avatar's surrounding or one or more sounds of an application can be rendered or generated by a sound renderer (i.e. Sound Renderer 96, etc.). Receiving comprises any action or operation by or for an Object 615, Collection of Object Representations 525, stream of Collections of Object Representations 525, Object Representation 625, Object Property 630, Object Processing Unit 140, Picture Renderer 91, Sound Renderer 96, Picture Recognizer 92, Sound Recognizer 97, and/or other disclosed elements.

At step 9110, a first one or more instruction sets for operating an avatar are received. In some aspects, an instruction set (i.e. Instruction Set 526, etc.) may be used or executed for operating an avatar (i.e. Avatar 605, etc.) of an application (i.e. Application Program 18, etc.). In other aspects, where an application does not comprise an avatar or rely on an avatar for its operation, an instruction set may be used or executed for operating the application. Therefore, a reference to an instruction set for operating an avatar includes or can be substituted with a reference to an instruction set for operating an application depending on context. In some embodiments, an instruction set can be used or executed by a processor (i.e. Processor 11, etc.) in operating an avatar and/or application. Operating an avatar and/or application includes performing or causing any operations on/by/with the avatar and/or application. In some designs, an instruction set can be received from an avatar, application, processor, and/or other processing element as the instruction set is being used or executed. In other designs, an instruction set can be received from an avatar, application, processor, and/or other processing element before or after the instruction set is used or executed. In further designs, an instruction set can be received from a running avatar, running application, running processor, and/or other running processing element. As such, an instruction set can be received at runtime. In some embodiments, receiving an instruction set includes tracing or profiling an avatar, application, processor, and/other processing elements. Tracing or profiling may include adding trace code or instrumentation to an avatar (i.e. avatar's object code, etc.) or application, and/or outputting trace information (i.e. instruction sets, etc.) to a receiving target. In some aspects, instrumentation can be performed in source code, bytecode, compiled code, interpreted code, translated code, machine code, and/or other code. In other aspects, instrumentation can be performed in various

elements of a computing system such as memory, virtual machine, runtime engine/environment, operating system, compiler, interpreter, translator, processor registers, and/or other elements. In yet other aspects, instrumentation can be performed in various abstraction layers of a computing system such as in software layer (i.e. application, etc.), in virtual machine (if VM is used), in operating system, in processor, and/or in other layers or areas that may exist in a particular computing system implementation. In yet other aspects, instrumentation can be performed at various times in an avatar's or application's execution such as source code write time, compile time, interpretation time, translation time, linking time, loading time, runtime, and/or others. In yet other aspects, instrumentation can be performed at various granularities or code segments such as some or all lines of code, some or all statements, some or all instructions or instruction sets, some or all basic blocks, some or all functions/routines/subroutines, and/or some or all other code segments. In yet other aspects, instrumentation may include a manual, automatic, dynamic, or just-in-time (JIT) instrumentation. In general, any instrumentation technique can be utilized. In further embodiments, receiving an instruction set includes attaching to or interfacing with an avatar, application, processor, and/or other processing element. In further embodiments, receiving an instruction set includes accessing and/or reading runtime engine/environment, virtual machine, operating system, compiler, interpreter, translator, execution stack, file, object, data structure, and/or other computing system elements. In further embodiments, receiving an instruction set includes accessing and/or reading memory, storage, and/or other repository. In further embodiments, receiving an instruction set includes accessing and/or reading processor registers and/or other processor elements. In further embodiments, receiving an instruction set includes accessing and/or reading inputs and/or outputs of an avatar, application, processor, and/or other processing element. In further embodiments, receiving an instruction set includes accessing and/or reading functions, methods, procedures, routines, subroutines, and/or other elements of an avatar and/or application. In further embodiments, receiving an instruction set includes accessing and/or reading source code, bytecode, compiled code, interpreted code, translated code, machine code, and/or other code. In further embodiments, receiving an instruction set includes accessing and/or reading values, variables, parameters, and/or other data or information. One or more instruction sets may temporally correspond to a collection of object representations. In general, one or more instruction sets that temporally correspond to a collection of object representations enable structuring knowledge of an avatar's and/or application's operation at or around the time of generating the collection of object representations. Such functionality enables spontaneous or seamless learning of an avatar's and/or application's operation in circumstances including objects with various properties as the avatar and/or application are operated in real time. Receiving comprises any action or operation by or for an Acquisition Interface 120, Instruction Set 526, and/or other disclosed elements.

At step 9115, the first collection of object representations is correlated with the first one or more instruction sets for operating the avatar. In some aspects, individual collections of object representations can be correlated with one or more instruction sets. In other aspects, streams of collections of object representations can be correlated with one or more instruction sets. In further aspects, individual collections of object representations or streams of collections of object representations can be correlated with the aforementioned temporally corresponding instruction sets. In further aspects, a collection of object representations or stream of collections of object representations may not be correlated with any instruction sets. Correlating may include structuring or generating a knowledge cell (i.e. Knowledge Cell 800, etc.) and storing one or more collections of object representations correlated with any instruction sets into the knowledge cell. Therefore, a knowledge cell may include any data structure or arrangement

that can facilitate such storing. A knowledge cell includes knowledge (i.e. unit of knowledge, etc.) of how an avatar and/or application operated in a circumstance including objects with various properties. In some designs, extra information (i.e. Extra Info 527, etc.) can optionally be used to facilitate enhanced comparisons or decision making in autonomous avatar and/or application operation where applicable. Therefore, any collection of object representations, instruction set, and/or other element may include or be correlated with extra information. Extra information may include any information useful in comparisons or decision making performed in autonomous avatar and/or application operation. Examples of extra information include time information, location information, computed information, visual information, acoustic information, contextual information, and/or other information. Correlating can be omitted where learning of an avatar's and/or application's operation in circumstances including objects with various properties is not implemented. Correlating comprises any action or operation by or for a Knowledge Structuring Unit 520, Knowledge Cell 800, and/or other disclosed elements.

At step 9120, the first collection of object representations correlated with the first one or more instruction sets for operating the avatar are stored. A collection of object representations correlated with one or more instruction sets may be part of a stored plurality of collections of object representations correlated with one or more instruction sets. Collections of object representations correlated with any instruction sets can be stored in a memory unit or other repository. The aforementioned knowledge cells comprising collections of object representations correlated with any instruction sets can be used in/as neurons, nodes, vertices, or other elements in any of the data structures or arrangements (i.e. neural network, graph, sequence, collection of knowledge cells, etc.) used for storing the knowledge of an avatar's and/or application's operation in circumstances including objects with various properties. Knowledge cells may be interconnected, interrelated, or interlinked into knowledge structures using statistical, artificial intelligence, machine learning, and/or other models or techniques. Such interconnected or interrelated knowledge cells can be used for enabling an avatar's and/or application's autonomous operation. The interconnected or interrelated knowledge cells may be stored or organized into a knowledgebase (i.e. Knowledgebase 530, etc.). In some embodiments, knowledgebase may be or include a neural network (i.e. Neural Network 530a, etc.). In other embodiments, knowledgebase may be or include a graph (i.e. Graph 530b, etc.). In further embodiments, knowledgebase may be or include a collection of sequences (i.e. Collection of Sequences 530c, etc.). In further embodiments, knowledgebase may be or include a sequence (i.e. Sequence 533, etc.). In further embodiments, knowledgebase may be or include a collection of knowledge cells (i.e. Collection of Knowledge Cells 530d, etc.). In general, knowledgebase may be or include any data structure or arrangement, and/or repository capable of storing the knowledge of an avatar's and/or application's operation in circumstances including objects with various properties. Knowledgebase may also include or be substituted with various artificial intelligence methods, systems, and/or models for knowledge structuring, storing, and/or representation such as deep learning, supervised learning, unsupervised learning, neural networks (i.e. convolutional neural network, recurrent neural network, deep neural network, etc.), search-based, logic and/or fuzzy logic-based, optimization-based, tree/graph/other data structure-based, hierarchical, symbolic and/or sub-symbolic, evolutionary, genetic, multi-agent, deterministic, probabilistic, statistical, and/or other methods, systems, and/or models. Storing can be omitted where learning of an avatar's and/or application's operation in circumstances including objects with various properties is not implemented. Storing comprises any action or operation by or for a Memory 12, Storage 27, Knowledgebase 530, Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells

530d, Knowledge Cell 800, Node 852, Connection 853, Layer 854, Similarity Comparison 125, and/or other disclosed elements.

At step 9125, a new collection of object representations is received. Step 9125 may include any action or operation described in Step 9105 as applicable.

At step 9130, the new collection of object representations is compared with the first collection of object representations. Comparing one collection of object representations with another collection of object representations may include comparing at least a portion of one collection of object representations with at least a portion of the other collection of object representations. In some embodiments, collections of object representations may be compared individually. In some aspects, comparing of individual collections of object representations may include comparing one or more object representations of one collection of object representations with one or more object representations of another collection of object representations. In other aspects, comparing of object representations may include comparing one or more object properties of one object representation with one or more object properties of another object representation. In some designs, one or more object properties in the same category (i.e. Category 635, etc.) can be compared. Comparing may include any techniques for comparing text, numbers, and/or other data. In further aspects, some object representations, object properties, and/or other elements of a collection of object representations can be omitted from comparison depending on implementation. In other embodiments, collections of object representations may be compared collectively as part of streams of collections of object representations. Collective comparing of collections of object representations may include any features, functionalities, and embodiments of the aforementioned individual comparing of collections of object representations. In some aspects, collective comparing of collections of object representations may include comparing one or more collections of object representations of one stream of collections of object representations with one or more collections of object representations of another stream of collections of object representations. In some designs, one or more corresponding (i.e. similarly ordered, temporally related, etc.) collections of object representations from the compared streams of collections of object representations can be compared. In other designs, Dynamic Time Warping (DTW) and/or other techniques can be utilized for comparison and/or aligning temporal sequences (i.e. streams of collections of object representations, etc.) that may vary in time or speed. In some aspects, some collections of object representations can be omitted from comparison depending on implementation. Any combination of the aforementioned and/or other elements or techniques can be utilized in alternate embodiments of the comparing. Comparing can be omitted where anticipating of an avatar's and/or application's operation in circumstances including objects with various properties is not implemented. Comparing comprises any action or operation by or for a Similarity Comparison 125, Decision-making Unit 540, and/or other disclosed elements.

At step 9135, a determination is made that there is at least a partial match between the new collection of object representations and the first collection of object representations. In some embodiments, determining at least a partial match between individually compared collections of object representations includes determining that a similarity between one or more portions of one collection of object representations and one or more portions of another collection of object representations exceeds a similarity threshold. In other embodiments, determining at least a partial match between individually compared collections of object representations includes determining at least a partial match between one or more portions of one collection of object representations and one or more

portions of another collection of object representations. In further embodiments, determining at least a partial match between individually compared collections of object representations includes determining substantial similarity between one or more portions of one collection of object representations and one or more portions of another collection of object representations. A portion of a collection of object representations may include an object representation, an object property, and/or other portion or element of the collection of object representations. In further embodiments, determining at least a partial match between individually compared collections of object representations includes determining that the number or percentage of matching or substantially matching object representations of the compared collections of object representations exceeds a threshold number (i.e. 1, 2, 4, 7, 18, etc.) or threshold percentage (i.e. 41%, 62%, 79%, 85%, 93%, etc.). In some aspects, type of object representations, importance of object representations, and/or other elements or techniques relating to object representations can be utilized for determining similarity using object representations. In further aspects, some of the object representations can be omitted in determining similarity using object representations depending on implementation. In further embodiments, determining a match or substantial match between compared object representations includes determining that the number or percentage of matching or substantially matching object properties of the compared object representations exceeds a threshold number (i.e. 1, 2, 3, 6, 11, etc.) or a threshold percentage (i.e. 55%, 61%, 78%, 82%, 99%, etc.). In some aspects, categories of object properties, importance of object properties, and/or other elements or techniques relating to object properties can be utilized for determining similarity using object properties. In further aspects, some of the object properties can be omitted in determining similarity using object properties depending on implementation. In some designs, substantial similarity of individually compared collections of object representations can be achieved when a similarity between one or more portions of one collection of object representations and one or more portions of another collection of object representations exceeds a similarity threshold. In other designs, substantial similarity of individually compared collections of object representations can be achieved when the number or percentage of matching or substantially matching object representations of the compared collections of object representations exceeds a threshold number (i.e. 1, 2, 4, 7, 18, etc.) or threshold percentage (i.e. 41%, 62%, 79%, 85%, 93%, etc.). In further aspects, substantial similarity of compared object representations can be achieved when the number or percentage of matching or substantially matching object properties of the compared object representations exceeds a threshold number (i.e. 1, 2, 3, 6, 11, etc.) or a threshold percentage (i.e. 55%, 61%, 78%, 82%, 99%, etc.). In some embodiments, determining at least a partial match between collectively compared collections of object representations (i.e. streams of collections of object representations, etc.) includes determining that the number or percentage of matching or substantially matching collections of object representations of the compared streams of collections of object representations exceeds a threshold number (i.e. 1, 2, 4, 9, 33, 138, etc.) or threshold percentage (i.e. 39%, 58%, 77%, 88%, 94%, etc.). In some aspects, importance of collections of object representations, order of collections of object representations, and/or other elements or techniques relating to collections of object representations can be utilized for determining similarity of collectively compared collections of object representations or streams of collections of object representations. In further aspects, some of the collections of object representations can be omitted in determining similarity of collectively compared collections of object representations or streams of collections of object representations depending on implementation. In some designs, a threshold for a number or percentage similarity can be used to determine a match or substantial match between any of the aforementioned

elements. Any text, number, and/or other data similarity determination techniques can be used in any of the aforementioned similarity determinations. A partial match of any of the compared elements may include a substantially or otherwise similar match, and vice versa. Therefore, these terms may be used interchangeably herein depending on context. Although, substantial similarity or substantial match is frequently used herein, it should be understood that any level of similarity, however high or low, may be utilized as defined by the rules (i.e. thresholds, etc.) for similarity. Any combination of the aforementioned and/or other elements or techniques can be utilized in alternate embodiments. Determining can be omitted where anticipating of an avatar's and/or application's operation in circumstances including objects with various properties is not implemented. Determining comprises any action or operation by or for a Similarity Comparison 125, Decision-making Unit 540, and/or other disclosed elements.

At step 9140, the first one or more instruction sets for operating the avatar correlated with the first collection of object representations are executed. An instruction set can be executed by a processor and/or other processing element. Executing can be performed in response to the aforementioned determining. In some aspects, an instruction set anticipated or determined to be used for autonomous operating of an avatar of an application may be executed. In other aspects, where an application does not comprise an avatar or rely on an avatar for its operation, an instruction set anticipated or determined to be used for autonomous operating of an application may be executed. Therefore, a reference to an instruction set to be used or executed for autonomous operating of an avatar includes or can be substituted with a reference to an instruction set to be used or executed for autonomous operating of an application depending on context. In some aspects, instruction sets anticipated or determined to be used or executed in an avatar's and/or application's autonomous operation may be referred to as anticipatory instruction sets, alternate instruction sets, and/or other suitable name or reference. Therefore, these terms can be used interchangeably herein depending on context. Executing may include executing one or more alternate instruction sets (i.e. anticipatory instruction sets, etc.) instead of or prior to an instruction set that would have been executed in a regular course of execution. In some embodiments, executing may include modifying a register or other element of a processor with one or more alternate instruction sets. Executing may also include redirecting a processor to one or more alternate instruction sets. In further embodiments, a processor may run an application including instruction sets for operating an avatar and/or the application. In some aspects, executing includes executing one or more alternate instruction sets as part of the avatar and/or application. In other aspects, executing includes modifying the avatar and/or application with one or more alternate instruction sets. In further aspects, executing includes redirecting the avatar and/or application to one or more alternate instruction sets. In further aspects, executing includes modifying one or more instruction sets of the avatar and/or application. In further aspects, executing includes modifying the avatar's and/or application's source code, bytecode, intermediate code, compiled code, interpreted code, translated code, runtime code, assembly code, machine code, or other code. In further aspects, executing includes modifying memory, processor register, storage, repository, and/or other elements where the avatar's and/or application's instruction sets are stored or used. In further aspects, executing includes modifying the avatar and/or application at source code write time, compile time, interpretation time, translation time, linking time, loading time, runtime, or other time. In further aspects, executing includes modifying one or more of the avatar's and/or application's lines of code, statements, instructions, functions, routines, subroutines, basic blocks, or other code segments. In further aspects, executing includes a manual, automatic, dynamic, just in time (JIT), or other instrumentation of the avatar and/or application. In further aspects, executing includes utilizing a dynamic, interpreted, scripting, or other programming

language. In further aspects, executing includes utilizing dynamic code, dynamic class loading, or reflection. In further aspects, executing includes utilizing assembly language. In further aspects, executing includes utilizing metaprogramming, self-modifying code, and/or a tool or technique for modifying the avatar and/or application. In further aspects, executing includes utilizing just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further aspects, executing includes utilizing dynamic expression creation, dynamic expression execution, dynamic function creation, or dynamic function execution. In further aspects, executing includes adding or inserting additional code into the avatar's and/or application's code. In further aspects, executing includes modifying, removing, rewriting, or overwriting the avatar's and/or application's code. In further aspects, executing includes branching, redirecting, extending, or hot swapping the avatar's and/or application's code. Branching or redirecting the avatar's and/or application's code may include inserting a branch, jump, or other means for redirecting the avatar's and/or application's execution. In further embodiments, executing includes modifying a virtual machine, a runtime engine, a compiler/interpreter/translator, an operating system, an execution stack, a storage, a memory, an input, and/or other elements of a computing system used in operating an avatar and/or application. Executing may be caused by ACAAO Unit 100, Artificial Intelligence Unit 110, Modification Interface 130, and/or other disclosed elements. Executing comprises any action or operation by or for a Processor 11, Application Program 18, Avatar 605, Modification Interface 130, and/or other disclosed elements.

At step 9145, one or more operations defined by the first one or more instruction sets for operating the avatar correlated with the first collection of object representations are performed by the avatar. The one or more operations may be performed in response to the aforementioned executing. In some aspects, an operation includes any operation that can be performed by/with/on an avatar. For example, an operation includes moving, maneuvering, jumping, running, shooting, selecting, and/or other operations in a context of a computer game, virtual world, 3D or 2D graphics application, and/or others. An operation of an avatar may include other operations in contexts of other applications. In further aspects, where an application does not comprise an avatar or rely on an avatar for its operation, an operation includes any operation that can be performed by/with/on an application and/or objects thereof. One of ordinary skill in art will recognize that, while all possible variations of operations by/with/on an avatar and/or application are too voluminous to list and limited only by the avatar's and/or application's design and/or user's utilization, other operations are within the scope of this disclosure.

Referring to Fig. 31, the illustration shows an embodiment of a method 9200 for learning and/or using an avatar's circumstances for autonomous avatar operation. In some aspects, the method can be used on a computing device or system to enable learning of an avatar's operation in circumstances including objects with various properties and enable autonomous avatar operation in similar circumstances. Method 9200 may include any action or operation of any of the disclosed methods such as method 9100, 9300, 9400, 9500, 9600, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method 9200.

At step 9205, a first collection of object representations is received. Step 9205 may include any action or operation described in Step 9105 of method 9100 as applicable.

At step 9210, a first one or more instruction sets for operating an avatar are received. Step 9210 may include any action or operation described in Step 9110 of method 9100 as applicable.

At step 9215, the first collection of object representations correlated with the first one or more instruction sets for operating the avatar are learned. Step 9215 may include any action or operation described in Step 9115 and/or Step 9120 of method 9100 as applicable.

At step 9220, a new collection of object representations is received. Step 9220 may include any action or operation described in Step 9125 of method 9100 as applicable.

At step 9225, the first one or more instruction sets for operating the avatar correlated with the first collection of object representations are anticipated based on at least a partial match between the new collection of object representations and the first collection of object representations. Step 9225 may include any action or operation described in Step 9130 and/or Step 9135 of method 9100 as applicable.

At step 9230, the first one or more instruction sets for operating the avatar correlated with the first collection of object representations are executed. Step 9230 may include any action or operation described in Step 9140 of method 9100 as applicable.

At step 9235, one or more operations defined by the first one or more instruction sets for operating the avatar correlated with the first collection of object representations are performed by the avatar. Step 9235 may include any action or operation described in Step 9145 of method 9100 as applicable.

Referring to Fig. 32, the illustration shows an embodiment of a method 9300 for learning and/or using an avatar's circumstances for autonomous avatar operation. In some aspects, the method can be used on a computing device or system to enable learning of an avatar's operation in circumstances including objects with various properties and enable autonomous avatar operation in similar circumstances. Method 9300 may include any action or operation of any of the disclosed methods such as method 9100, 9200, 9400, 9500, 9600, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method 9300.

At step 9305, a first stream of collections of object representations is received. Step 9305 may include any action or operation described in Step 9105 of method 9100 as applicable.

At step 9310, a first one or more instruction sets for operating an avatar are received. Step 9310 may include any action or operation described in Step 9110 of method 9100 as applicable.

At step 9315, the first stream of collections of object representations is correlated with the first one or more instruction sets for operating the avatar. Step 9315 may include any action or operation described in Step 9115 of method 9100 as applicable.

At step 9320, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the avatar are stored. Step 9320 may include any action or operation described in Step 9120 of method 9100 as applicable.

At step 9325, a new stream of collections of object representations is received. Step 9325 may include any action or operation described in Step 9125 of method 9100 as applicable.

At step 9330, the new stream of collections of object representations is compared with the first stream of collections of object representations. Step 9330 may include any action or operation described in Step 9130 of method 9100 as applicable.

At step 9335, a determination is made that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. Step 9335 may include any action or operation described in Step 9135 of method 9100 as applicable.

At step 9340, the first one or more instruction sets for operating the avatar correlated with the first stream of collections of object representations are executed. Step 9340 may include any action or operation described in Step 9140 of method 9100 as applicable.

At step 9345, one or more operations defined by the first one or more instruction sets for operating the avatar correlated with the first stream of collections of object representations are performed by the avatar. Step 9345 may include any action or operation described in Step 9145 of method 9100 as applicable.

Referring to Fig. 33, the illustration shows an embodiment of a method 9400 for learning and/or using an application's circumstances for autonomous application operation. In some aspects, the method can be used on a computing device or system to enable learning of an application's operation in circumstances including objects with various properties and enable autonomous application operation in similar circumstances. Method 9400 may include any action or operation of any of the disclosed methods such as method 9100, 9200, 9300, 9500, 9600, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method 9400.

At step 9405, a first collection of object representations is received. Step 9405 may include any action or operation described in Step 9105 of method 9100 as applicable.

At step 9410, a first one or more instruction sets for operating an application are received. Step 9410 may include any action or operation described in Step 9110 of method 9100 as applicable.

At step 9415, the first collection of object representations is correlated with the first one or more instruction sets for operating the application. Step 9415 may include any action or operation described in Step 9115 of method 9100 as applicable.

At step 9420, the first collection of object representations correlated with the first one or more instruction sets for operating the application are stored. Step 9420 may include any action or operation described in Step 9120 of method 9100 as applicable.

At step 9425, a new collection of object representations is received. Step 9425 may include any action or operation described in Step 9125 of method 9100 as applicable.

At step 9430, the new collection of object representations is compared with the first collection of object representations. Step 9430 may include any action or operation described in Step 9130 of method 9100 as applicable.

At step 9435, a determination is made that there is at least a partial match between the new collection of object representations and the first collection of object representations. Step 9435 may include any action or operation described in Step 9135 of method 9100 as applicable.

At step 9440, the first one or more instruction sets for operating the application correlated with the first collection of object representations are executed. Step 9440 may include any action or operation described in Step 9140 of method 9100 as applicable.

At step 9445, one or more operations defined by the first one or more instruction sets for operating the application correlated with the first collection of object representations are performed by the application. Step 9445 may include any action or operation described in Step 9145 of method 9100 as applicable.

Referring to Fig. 34, the illustration shows an embodiment of a method 9500 for learning and/or using an application's circumstances for autonomous application operation. In some aspects, the method can be used on a computing device or system to enable learning of an application's operation in circumstances including objects with various properties and enable autonomous application operation in similar circumstances. Method 9500 may include any action or operation of any of the disclosed methods such as method 9100, 9200, 9300, 9400, 9600, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method 9500.

At step 9505, a first collection of object representations is received. Step 9505 may include any action or operation described in Step 9105 of method 9100 as applicable.

At step 9510, a first one or more instruction sets for operating an application are received. Step 9510 may include any action or operation described in Step 9110 of method 9100 as applicable.

At step 9515, the first collection of object representations correlated with the first one or more instruction sets for operating the application are learned. Step 9515 may include any action or operation described in Step 9115 and/or Step 9120 of method 9100 as applicable.

At step 9520, a new collection of object representations is received. Step 9520 may include any action or operation described in Step 9125 of method 9100 as applicable.

At step 9525, the first one or more instruction sets for operating the application correlated with the first collection of object representations are anticipated based on at least a partial match between the new collection of object representations and the first collection of object representations. Step 9525 may include any action or operation described in Step 9130 and/or Step 9135 of method 9100 as applicable.

At step 9530, the first one or more instruction sets for operating the application correlated with the first collection of object representations are executed. Step 9530 may include any action or operation described in Step 9140 of method 9100 as applicable.

At step 9535, one or more operations defined by the first one or more instruction sets for operating the application correlated with the first collection of object representations are performed by the application. Step 9535 may include any action or operation described in Step 9145 of method 9100 as applicable.

Referring to Fig. 35, the illustration shows an embodiment of a method 9600 for learning and/or using an application's circumstances for autonomous application operation. In some aspects, the method can be used on a computing device or system to enable learning of an application's operation in circumstances including objects with various properties and enable autonomous application operation in similar circumstances. Method 9600 may include any action or operation of any of the disclosed methods such as method 9100, 9200, 9300, 9400, 9500, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method 9600.

At step 9605, a first stream of collections of object representations is received. Step 9605 may include any action or operation described in Step 9105 of method 9100 as applicable.

At step 9610, a first one or more instruction sets for operating an application are received. Step 9610 may include any action or operation described in Step 9110 of method 9100 as applicable.

At step 9615, the first stream of collections of object representations is correlated with the first one or more instruction sets for operating the application. Step 9615 may include any action or operation described in Step 9115 of method 9100 as applicable.

At step 9620, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the application are stored. Step 9620 may include any action or operation described in Step 9120 of method 9100 as applicable.

At step 9625, a new stream of collections of object representations is received. Step 9625 may include any action or operation described in Step 9125 of method 9100 as applicable.

At step 9630, the new stream of collections of object representations is compared with the first stream of collections of object representations. Step 9630 may include any action or operation described in Step 9130 of method 9100 as applicable.

At step 9635, a determination is made that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. Step 9635 may include any action or operation described in Step 9135 of method 9100 as applicable.

At step 9640, the first one or more instruction sets for operating the application correlated with the first stream of collections of object representations are executed. Step 9640 may include any action or operation described in Step 9140 of method 9100 as applicable.

At step 9645, one or more operations defined by the first one or more instruction sets for operating the application correlated with the first stream of collections of object representations are performed by the application. Step 9645 may include any action or operation described in Step 9145 of method 9100 as applicable.

Referring to Fig. 36, in some exemplary embodiments, Application Program 18 may be or include a 3D Computer Game 18a. Examples of 3D Computer Game 18a include a first shooter game, a flight simulation, a driving simulation, and/or others. Avatar 605 may be or include Soldier 605a within 3D Computer Game 18a. Soldier 605a can be controlled by User 50 (i.e. game player, etc.) through inputting operating directions via Human-machine Interface 23 such as a game controller, keyboard, joystick, or other input device. For instance, responsive to User's 50 manipulating one or more game controller elements, Soldier 605a may be caused to move, maneuver, shoot, jump, and/or perform other operations. Computing Device 70 may include or be coupled to ACAAO Unit 100. ACAAO Unit 100 can obtain objects (i.e. Opponent 615aa, Dog 615ab, Pedestrian 615ac, etc.) and/or their properties in Soldier's 605a surrounding within 3D Computer Game 18a. ACAAO Unit 100 may create or generate one or more (i.e. stream, etc.) Collections of Object Representations 525 comprising Object Representations 625, Object Properties 630, and/or other elements or information representing Objects 615 in Soldier's 605a surrounding. ACAAO Unit 100 can also obtain Instruction Sets 526 used or executed in operating Soldier 605a. ACAAO Unit 100 can also optionally obtain any Extra Info 527 (i.e. time, location, computed, visual, acoustic, contextual, and/or other information, etc.) related to Soldier's 605a operation. As User 50 operates Soldier 605a in circumstances including objects with various properties as shown, ACAAO Unit 100 may learn Soldier's 605a operation in these

circumstances by correlating Collections of Object Representations 525 representing Objects 615 in Soldier's 605a surrounding with one or more Instruction Sets 526 used or executed in operating Soldier 605a. Any Extra Info 527 related to Soldier's 605a operation may also optionally be correlated with Collections of Object Representations 525. ACAAO Unit 100 can store this knowledge into Knowledgebase 530 (i.e. Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.). In the future, ACAAO Unit 110 may compare incoming Collections of Object Representations 525 representing Objects 615 in Soldier's 605a surrounding with previously learned Collections of Object Representations 525 including optionally using any Extra Info 527 for enhanced decision making. If substantially similar or at least a partial match is found or determined, ACAAO Unit 110 may cause the Instruction Sets 526 correlated with the previously learned Collections of Object Representations 525 to be executed, thereby enabling autonomous operation of Soldier 605a in similar circumstances as in previously learned ones. For instance, ACAAO Unit 100 may learn User 50-directed shooting at Opponent 615aa by Soldier 605a in a circumstance that includes Opponent 615aa, Dog 615ab, Pedestrian 615ac, and/or other Objects 615 among which Soldier 605a may need to maneuver and/or with which Soldier 605a may need to interact, as shown. In the future, when a circumstance that includes one or more Objects 615 with similar Object Properties 630 is encountered, ACAAO Unit 100 may implement the shooting at Opponent 615aa by Soldier 605a autonomously. In some designs, 3D Computer Game 18a may include elevated Objects 615 such as flying objects (i.e. flying animals, aircraft, etc.), objects on hills or mountains, objects on buildings, and/or others in which case altitudinal information related to distance and bearing/angle of Objects 615 relative to Soldier 605a can be obtained, learned, and used. In other designs, the street (not enumerated), parts thereof (i.e. curbs, sidewalks, etc.), and/or objects thereon (i.e. buildings, etc.) can be obtained as Objects 615 themselves, which may be learned and used.

In some embodiments, ACAAO Unit 100 may reside on Server 96 accessible over Network 95 as previously described. In such embodiments, any number of Computing Devices 70, Processors 11, 3D Computer Games 18a, and/or other elements may connect to such remote ACAAO Unit 100 and the remote ACAAO Unit 100 may learn operations of their Soldiers 605a in circumstances including objects with various properties. In turn, any number of Computing Devices 70, Processors 11, 3D Computer Games 18a, and/or other elements can utilize the remote ACAAO Unit 100 for autonomous operation of their Soldiers 605a. For example, multiple Users 50 (i.e. game players, etc.) may operate their Soldiers 605a in 3D Computer Game 18a running on their respective Computing Devices 70 where the Computing Devices 70 and/or elements thereof may be configured to transmit Soldiers' 605a operations in circumstances including objects with various properties to a remote ACAAO Unit 100. Such remote ACAAO Unit 100 enables learning of the game players' collective knowledge of operating Soldier 605a in circumstances including objects with various properties. Any of the disclosed elements such as Artificial Intelligence Unit 110, Knowledgebase 530, and/or others can reside on Server 96, and any combination of local and remote elements can be implemented in alternate embodiments.

In some embodiments, a combination of ACAAO Unit 100 and other systems and/or techniques can be utilized to implement Soldier's 605a operation. In one example, ACAAO Unit 100 can be a primary or preferred system for implementing Soldier's 605a operation. While operating autonomously under the control of ACAAO Unit 100, Soldier 605a may encounter a circumstance including objects with various properties that has not been encountered or learned before. In such situations, User 50 and/or non-ACAAO system may take control of Soldier's

605a operation. ACAAO Unit 100 may take control again when Soldier 605a encounters a previously learned circumstance including objects with various properties. Naturally, ACAAO Unit 100 can learn Soldier's 605a operation in circumstances while User 50 and/or non-ACAAO system is in control of Soldier 605a, thereby reducing or eliminating the need for future involvement of User 50 and/or non-ACAAO system. In another example, User 50 and/or non-ACAAO system can be a primary or preferred system for implementing Soldier's 605a operation. While operating under the control of User 50 and/or non-ACAAO system, User 50 and/or non-ACAAO system may release control to ACAAO Unit 100 for any reason (i.e. User 50 gets tired or distracted, non-ACAAO system gets stuck or cannot make a decision, etc.), at which point Soldier 605a can be controlled by ACAAO Unit 100. In some designs, ACAAO Unit 100 may take control in certain special circumstances including objects with various properties where ACAAO Unit 100 offers superior performance even though User 50 and/or non-ACAAO system may generally be preferred. Once Soldier 605a leaves such special circumstances, ACAAO Unit 100 may release control to User 50 and/or non-ACAAO system. In general, ACAAO Unit 100 can take control from, share control with, or release control to User 50, non-ACAAO system, and/or other system or process at any time, in any circumstances, and remain in control for any period of time as needed.

In some embodiments, ACAAO Unit 100 may control one or more elements of Soldier 605a while User 50 and/or non-ACAAO system may control other one or more elements of Soldier 605a. For example, ACAAO Unit 100 may control Soldier's 605a movement, while User 50 and/or non-ACAAO system may control Soldier's 605a aiming and shooting. Any other combination of controlling various elements or functions of Soldier 605a by ACAAO Unit 100, User 50, and/or non-ACAAO system can be implemented.

In some embodiments, ACAAO Unit 100 enables learning of a particular User's 50 (i.e. game player's, etc.) knowledge, methodology, or style of operating Soldier 605a within 3D Game Application 18a. In some aspects, learning of a particular User's 50 knowledge, methodology, or style of operating Soldier 605a includes learning the User's 50 directing or operating Soldier 605a in circumstances including objects with various properties. In one example, one User 50 may shoot an opponent while another User 50 may strike the opponent with a sword. In another example, one User 50 may jump over an obstacle while another User 50 may move around the obstacle, and so on. The knowledge of User's 50 methodology or style of operating Soldier 605a can be used to enable personalized autonomous operation of Soldier 605a specific to a particular User 50. Therefore, ACAAO-enabled Soldier 605a can exemplify User's 50 knowledge, methodology, or style of operating Soldier 605a as learned from User 50. In some aspects, this functionality enables one or more ACAAO-enabled Soldiers 605a to be utilized in 3D Computer Game 18a to assist User 50 in defeating an opponent or achieving other goals. In one example, User 50 can utilize a team of ACAAO-enabled Soldiers 605a each of which may exemplify User's 50 knowledge, methodology, or style of operating Soldier 605a. In one instance, ACAAO-enabled Soldiers 605a may be dispersed around a User 50-controlled Soldier 605a within a specific radius and follow User 50-controlled Soldier's 605a movement. In another instance, ACAAO-enabled Soldiers 605a may move autonomously toward a certain point or goal in 3D Computer Game 18a. In a further instance, ACAAO-enabled Soldiers 605a can be completely autonomous and rely solely on the knowledge learned from User's 50 methodology or style of operating Soldier 605a. In other aspects, ACAAO Unit 100 enables a professional or other experienced User 50 (i.e. game player, etc.) to record his/her knowledge, methodology, or style of operating Soldier 605a into Knowledgebase 530 (i.e. Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells

530d, etc.) and/or other repository. User 50 can then sell or make available his/her knowledge, methodology, or style of operating Soldier 605a to other users who may want to implement User's 50 knowledge, methodology, or style of operating Soldier 605a. Knowledgebase 530 and/or other repository comprising User's 50 knowledge, methodology, or style of operating Soldier 605a can be available to other users via a storage medium, via a network, or via other means.

One of ordinary skill in art will understand that the aforementioned features, functionalities, and embodiments described with respect to 3D Computer Game 18a can be implemented in any 3D Application Program 18 such as a 3D virtual world, 3D graphics application, computer aided design (CAD) application, and/or others. Similar features, functionalities, and embodiments can also be implemented in a 2D Application Program 18, and/other application program as applicable, and vice versa.

Referring to Fig. 37, in some exemplary embodiments, Application Program 18 may be or include a 2D Computer Game 18b. Examples of 2D Computer Game 18b include a strategy game, a shooter game, a tile-matching game, a platform game, and/or others. Avatar 605 may be or include Tank 605b within 2D Computer Game 18b. Tank 605b can be controlled by User 50 (i.e. game player, etc.) through inputting operating directions via Human-machine Interface 23 such as a game controller, keyboard, joystick, or other input device. For instance, responsive to User's 50 manipulating one or more game controller elements, Tank 605b may be caused to move, maneuver, shoot, and/or perform other operations. Computing Device 70 may include or be coupled to ACAAO Unit 100. ACAAO Unit 100 can obtain objects (i.e. Helicopter 615ba, Rocket Launcher 615bb, Communication Center 615bc, Tank 615bd, etc.) and/or their properties in Tank's 605b surrounding within 2D Computer Game 18b. ACAAO Unit 100 may create or generate one or more (i.e. stream, etc.) Collections of Object Representations 525 comprising Object Representations 625, Object Properties 630, and/or other elements or information representing Objects 615 in Tank's 605b surrounding. ACAAO Unit 100 can also obtain Instruction Sets 526 used or executed in operating Tank 605b. ACAAO Unit 100 can also optionally obtain any Extra Info 527 (i.e. time, location, computed, visual, acoustic, contextual, and/or other information, etc.) related to Tank's 605b operation. As User 50 operates Tank 605b in circumstances including objects with various properties as shown, ACAAO Unit 100 may learn Tank's 605b operation in these circumstances by correlating Collections of Object Representations 525 representing Objects 615 in Tank's 605b surrounding with one or more Instruction Sets 526 used or executed in operating Tank 605b. Any Extra Info 527 related to Tank's 605b operation may also optionally be correlated with Collections of Object Representations 525. ACAAO Unit 100 can store this knowledge into Knowledgebase 530 (i.e. Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.). In the future, ACAAO Unit 110 may compare incoming Collections of Object Representations 525 representing Objects 615 in Tank's 605b surrounding with previously learned Collections of Object Representations 525 including optionally using any Extra Info 527 for enhanced decision making. If substantially similar or at least a partial match is found or determined, ACAAO Unit 110 may cause the Instruction Sets 526 correlated with the previously learned Collections of Object Representations 525 to be executed, thereby enabling autonomous operation of Tank 605b in similar circumstances as in previously learned ones. For instance, ACAAO Unit 100 may learn User 50-directed maneuvering and shooting by Tank 605b in a circumstance that includes Helicopter 615ba, Rocket Launcher 615bb, Communication Center 615bc, Tank 615bd, and/or other Objects 615 among which Tank 605b may need to maneuver and/or with which Tank 605b may need to interact, as shown. In the future, when a circumstance that

includes one or more Objects 615 with similar Object Properties 630 is encountered, ACAAO Unit 100 may implement the maneuvering and shooting by Tank 605b autonomously. In some designs, 2D Computer Game 18b may include 3D effects such as the effect of flying objects (i.e. flying animals, aircraft, Helicopter 615ba, etc.), objects on hills or mountains, objects on buildings, and/or other elevated objects in which case altitudinal information related to distance and bearing/angle of Objects 615 relative to Tank 605b can be obtained, learned, and used. In other designs, the canyon (not enumerated) and/or parts thereof (i.e. cliffs, ridges, etc.) can be obtained as Objects 615 themselves, which may be learned and used.

Referring to Fig. 38, in some exemplary embodiments, an Area of Interest 450 can be utilized. In one example, Area of Interest 450 may include a radial, circular, elliptical, or other such area around Tank 605b. In another example, Area of Interest 450 may include a triangular, rectangular, octagonal, or other such area around Tank 605b. In a further example, Area of Interest 450 may include a spherical, cubical, pyramid-like, or other such area around Tank 605b as applicable to 3D space. Any other Area of Interest 450 shape can be utilized depending on implementation. The shape and/or size of Area of Interest 450 can be defined by a user, by ACAAO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. For instance, Area of Interest 450 may include or be defined by a circle around Tank 605b with a radius of 250 meters. Any other radiuses or sizes of Area of Interest 450 can be used such as 0.27m, 1m, 7m, 19m, 382m, 7116m, 49276m, and so on. Utilizing Area of Interest 450 enables ACAAO Unit 100 to focus on Tank's 605b surrounding, thereby ignoring extraneous detail in the rest of the space. In some designs, Tank's 605b surrounding may include or be defined by Area of Interest 450. In some aspects, Area of Interest 450 can be subdivided into sub-areas (i.e. sub-circles, sub-rectangles, sub-spheres, etc.). Sub-areas can be used to classify the surrounding by distance from Tank 605b. For example, the surrounding closer to Tank 605b may be more important and may be assigned higher importance index or weight. As User 50 operates Tank 605b in circumstances including objects with various properties as shown, ACAAO Unit 100 may learn Tank's 605b operation in these circumstances by correlating Collections of Object Representations 525 representing Objects 615 in Area of Interest 450 around Tank 605b with one or more Instruction Sets 526 used or executed in operating Tank 605b. Any Extra Info 527 related to Tank's 605b operation may also optionally be correlated with Collections of Object Representations 525. ACAAO Unit 100 can store this knowledge into Knowledgebase 530 (i.e. Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.). In the future, ACAAO Unit 110 may compare incoming Collections of Object Representations 525 representing Objects 615 in Area of Interest 450 around Tank 605b with previously learned Collections of Object Representations 525 including optionally using any Extra Info 527 for enhanced decision making. If substantially similar or at least a partial match is found or determined, ACAAO Unit 110 may cause the Instruction Sets 526 correlated with the previously learned Collections of Object Representations 525 to be executed, thereby enabling autonomous operation of Tank 605b in similar Areas of Interest 450 as in previously learned ones. For instance, ACAAO Unit 100 may learn User 50-directed maneuvering and shooting by Tank 605b in an Area of Interest 450 that includes Rocket Launcher 615bb, Communication Center 615bc, Tank 615bd, and/or other Objects 615 among which Tank 605b may need to maneuver and/or with which Tank 605b may need to interact, as shown. In the future, when an Area of Interest 450 that includes one or more Objects 615 with similar Object Properties 630 is encountered, ACAAO Unit 100 may implement the maneuvering and shooting by Tank 605b autonomously.

Referring to Fig. 39, in some exemplary embodiments, Application Program 18 may be or include a 2D Computer Game 18c comprising multiple Avatars 605 that User 50 can control or operate. As multiple Avatars 605 can perform operations in such 2D Computer Game 18c, Area of Interest 450 in 2D Computer Game 18c including multiple Avatars 605 and/or other Objects 615 may be more relevant than a particular Avatar's 605 surrounding or Area of Interest 450 for learning circumstances including objects with various properties. In some aspects, Area of Interest 450 in 2D Computer Game 18c may include the entire 2D Computer Game 18c, a part of 2D Computer Game 18c that is shown to User 50 (i.e. on a display, via a graphical user interface, etc.), or any part of 2D Computer Game 18c. Examples of 2D Computer Game 18c include a strategy game, a board game, a tile-matching game, and/or others. Avatars 605 may be or include Ships 605c1-605c3, etc. within 2D Computer Game 18c. Ships 605c1-605c3, etc. can be controlled by User 50 (i.e. game player, etc.) through inputting operating directions via Human-machine Interface 23 such as a game controller, keyboard, joystick, or other input device. For instance, responsive to User's 50 manipulating one or more game controller elements, Ships 605c1-605c3, etc. may be caused to move, maneuver, shoot, and/or perform other operations. Computing Device 70 may include or be coupled to ACAAO Unit 100. ACAAO Unit 100 can obtain objects and/or their properties in Area of Interest 450 in 2D Computer Game 18c. For instance, Area of Interest 450 in 2D Computer Game 18c may include Ships 605c1-605c3, etc., Ships 615c1-615c3, etc., and/or other objects. Properties of each of the objects may include a location defined by coordinates. ACAAO Unit 100 may create or generate one or more (i.e. stream, etc.) Collections of Object Representations 525 comprising Object Representations 625, Object Properties 630, and/or other elements or information representing Objects 615 in Area of Interest 450 in 2D Computer Game 18c. ACAAO Unit 100 can also obtain Instruction Sets 526 used or executed in operating Ships 605c1-605c3, etc. ACAAO Unit 100 can also optionally obtain any Extra Info 527 (i.e. time, location, computed, visual, acoustic, contextual, and/or other information, etc.) related to Ships' 605c1-605c3, etc. operation. As User 50 operates Ships 605c1-605c3, etc. in circumstances including objects with various properties as shown, ACAAO Unit 100 may learn Ships' 605c1-605c3, etc. operation in these circumstances by correlating Collections of Object Representations 525 representing Objects 615 in Area of Interest 450 in 2D Computer Game 18c with one or more Instruction Sets 526 used or executed in operating Ships 605c1-605c3, etc. Any Extra Info 527 related to Ships' 605c1-605c3, etc. operation may also optionally be correlated with Collections of Object Representations 525. ACAAO Unit 100 can store this knowledge into Knowledgebase 530 (i.e. Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.). In the future, ACAAO Unit 110 may compare incoming Collections of Object Representations 525 representing Objects 615 in Area of Interest 450 in 2D Computer Game 18c with previously learned Collections of Object Representations 525 including optionally using any Extra Info 527 for enhanced decision making. If substantially similar or at least a partial match is found or determined, ACAAO Unit 110 may cause the Instruction Sets 526 correlated with the previously learned Collections of Object Representations 525 to be executed, thereby enabling autonomous operation of Ships 605c1-605c3, etc. in similar circumstances as in previously learned ones. For instance, ACAAO Unit 100 may learn User 50-directed maneuvering and shooting by Ships 605c1-605c3, etc. in a circumstance that includes Ships 615c1-615c3 and/or other Objects 615 among which Ships 605c1-605c3, etc. may need to maneuver and/or with which Ships 605c1-605c3, etc. may need to interact, as shown. In the future, when a circumstance that includes one or more Objects 615 with similar Object Properties 630 is encountered, ACAAO Unit 100 may implement the maneuvering and shooting by Ships 605c1-605c3, etc.

autonomously. In some designs, the islands (not enumerated), parts thereof (i.e. cliffs, ridges, beaches, mountains, etc.), and/or objects thereon (i.e. buildings, etc.) can be obtained as Objects 615 themselves, which may be learned and used.

One of ordinary skill in art will understand that the features, functionalities, and embodiments described in the above exemplary embodiments with respect to Soldier 605a, Tank 605b, and Ships 605c1-605c3 can similarly be implemented by/with/on any Avatar 605 of Application Program 18 or by/with/on any Application Program 18.

In some exemplary embodiments (not depicted), Application Program 18 may not comprise an Avatar 605 and/or may not rely on Avatar 605 for its operation. Application Program 18 can be controlled by User 50 through inputting operating directions via Human-machine Interface 23 such as a mouse, keyboard, or other input device. For instance, responsive to User's 50 pressing one or more mouse buttons, moving the mouse, and/or pressing keyboard buttons, an Object 615 of Application Program 18 may be inserted, deleted, selected, moved, and/or subject to other operations. Computing Device 70 may include or be coupled to ACAAO Unit 100. ACAAO Unit 100 can obtain objects and/or their properties in Application Program 18. In some aspects, Area of Interest 450 in Application Program 18 can be utilized as previously described. Area of Interest 450 in Application Program 18 may include the entire Application Program 18, a part of Application Program 18 that is shown to User 50 (i.e. on a display, via a graphical user interface, etc.), or any part of Application Program 18. For instance, ACAAO Unit 100 can obtain Objects 615 in Application Program 18 shown to User 50 via a graphical user interface. ACAAO Unit 100 may create or generate one or more (i.e. stream, etc.) Collections of Object Representations 525 comprising Object Representations 625, Object Properties 630, and/or other elements or information representing Objects 615 in Application Program 18. ACAAO Unit 100 can also obtain Instruction Sets 526 used or executed in operating Application Program 18 and/or Objects 615 thereof. ACAAO Unit 100 can also optionally obtain any Extra Info 527 (i.e. time, location, computed, visual, acoustic, contextual, and/or other information, etc.) related to Application Program's 18 and/or Objects' 615 thereof operation. As User 50 operates Application Program 18 in circumstances including objects with various properties, ACAAO Unit 100 may learn Application Program's 18 operation in these circumstances by correlating Collections of Object Representations 525 representing Objects 615 in Application Program 18 with one or more Instruction Sets 526 used or executed in operating Application Program 18 and/or Objects 615 thereof. Any Extra Info 527 related to Application Program's 18 and/or Objects' 615 thereof operation may also optionally be correlated with Collections of Object Representations 525. ACAAO Unit 100 can store this knowledge into Knowledgebase 530 (i.e. Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.). In the future, ACAAO Unit 110 may compare incoming Collections of Object Representations 525 representing Objects 615 in Application Program 18 with previously learned Collections of Object Representations 525 including optionally using any Extra Info 527 for enhanced decision making. If substantially similar or at least a partial match is found or determined, ACAAO Unit 110 may cause the Instruction Sets 526 correlated with the previously learned Collections of Object Representations 525 to be executed, thereby enabling autonomous operation of Application Program 18 and/or Objects 615 thereof in similar circumstances as in previously learned ones. For instance, ACAAO Unit 100 may learn User 50-directed selecting an Object 615, moving an Object 615, and/or deleting an Object 615 of Application Program 18 in a circumstance that includes various Objects 615. In the future, when a circumstance that includes one or more

Objects 615 with similar Object Properties 630 is encountered, ACAAO Unit 100 may implement the selecting an Object 615, moving an Object 615, and/or deleting an Object 615 of Application Program 18 autonomously.

One of ordinary skill in art will understand that the functionalities described with respect to Application Program 18 and/or Objects 615 thereof can be implemented in any Application Program 18 such as a computer game, a virtual world, a 2D or 3D graphics application, a web browser, a media application, a word processing application, a spreadsheet application, a database application, a forms-based application, an operating system, a device/system control application, and/or others as applicable. In such Application Programs 18, examples of Objects 615 that can be utilized include a 2D model, a 3D model, a 2D shape (i.e. point, line, square, rectangle, circle, triangle, etc.), a 3D shape (i.e. cube, sphere, etc.), a graphical user interface (GUI) element, a form element (i.e. text field, radio button, push button, check box, etc.), a data or database element, a spreadsheet element, a link, a picture, a text (i.e. character, word, etc.), a number, and/or others as applicable. Application Program 18 may be or include any Application Program 18 that can benefit from the functionalities described herein.

It must be noted that as used herein and in the appended claims, the singular forms "a", "an", and "the" include plural referents unless the context clearly dictates otherwise.

A number of embodiments have been described herein. While this disclosure contains many specific implementation details, these should not be construed as limitations on the scope of the disclosure or of what may be claimed, but rather as descriptions of features specific to particular embodiments. It should be understood that various modifications can be made without departing from the spirit and scope of the disclosure. The logic flows depicted in the figures do not require the particular order shown, or sequential order, to achieve desirable results. Other or additional steps, elements, or connections can be included, or some of the steps, elements, or connections can be excluded, or a combination thereof can be utilized in the described flows, illustrations, or descriptions. Further, the various aspects of the disclosed devices, apparatuses, systems, and/or methods can be combined in whole or in part with each other to produce additional implementations. Moreover, separation of various components in the embodiments described herein should not be understood as requiring such separation in all embodiments, and it should be understood that the described components can generally be integrated together in a single product or packaged into multiple products. Accordingly, other embodiments are within the scope of the following claims.

Electronic Patent Application Fee Transmittal

Application Number:	
Filing Date:	
Title of Invention:	Your fee transmittal sheet cannot be generated at this time. Please complete your submission. After completing your submission, please contact the Electronic Business Center (EBC) to request the fee transmittal sheet for this submission. We apologize for the inconvenience. Electronic Business Center (EBC) 1-866-217-9197 (toll-free) or 571-272-4100 6 a.m. to 12 Midnight Eastern Time Monday -Friday
First Named Inventor/Applicant Name:	
Filer:	
Attorney Docket Number:	

Filing Fees

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Basic Filing:				
Pages:				
Claims:				
Miscellaneous-Filing:				
Petition:				
Patent-Appeals-and-Interference:				
Post-Allowance-and-Post-Issuance:				
Extension-of-Time:				

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Miscellaneous:				
	Total in USD ($)			

DocCode - SCORE

SCORE Placeholder Sheet for IFW Content

Application Number: 15382743 Document Date: 12/19/2016

The presence of this form in the IFW record indicates that the following document type was received in electronic format on the date identified above. This content is stored in the SCORE database.

- Drawings – Other than Black and White Line Drawings

Since this was an electronic submission, there is no physical artifact folder, no artifact folder is recorded in PALM, and no paper documents or physical media exist. The TIFF images in the IFW record were created from the original documents that are stored in SCORE.

To access the documents in the SCORE database, refer to instructions below.

At the time of document entry (noted above):

- Examiners may access SCORE content via the eDAN interface.
- Other USPTO employees can bookmark the current SCORE URL (http://Score.uspto.gov/ScoreAccessWeb/).
- External customers may access SCORE content via the Public and Private PAIR interfaces.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS Remove

Examiner Initial*	Cite No	Patent Number	Kind Code[1]	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	4370707		1983-01-25	Phillips , et al.	
	2	4730315		1988-03-08	Saito , et al.	
	3	4860203		1989-08-22	Corrigan , et al.	
	4	5602982		1997-02-11	Judd , et al.	
	5	6026234		2000-02-15	Hanson , et al.	
	6	6088731		2000-07-11	Kiraly , et al.	
	7	6106299		2000-08-22	Ackermann , et al.	
	8	6126330		2000-10-03	Knight	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	9	6314558		2001-11-06	Angel , et al.	
	10	6643842		2003-11-04	Angel , et al.	
	11	6728689		2004-04-27	Drissi , et al.	
	12	6735632		2004-05-11	Kiraly , et al.	
	13	6801912		2004-10-05	Moskowitz , et al.	
	14	6915105		2005-07-05	Masuda	
	15	6973446		2005-12-06	Mamitsuka , et al.	
	16	7017153		2006-03-21	Gouriou , et al.	
	17	7052277		2006-05-30	Kellman	
	18	7222127		2007-05-22	Bem , et al.	
	19	7240335		2007-07-03	Angel , et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	20	7249349		2007-07-24	Hundt , et al.	
	21	7424705		2008-09-09	Lewis , et al.	
	22	7478371		2009-01-13	Gove	
	23	7484205		2009-01-27	Venkatapathy	
	24	7721218		2010-05-18	Awe , et al.	
	25	7765537		2010-07-27	Havin , et al.	
	26	7797259		2010-09-14	Jiang , et al.	
	27	7840060		2010-11-23	Podilchuk	
	28	7925984		2011-04-12	Awe , et al.	
	29	7987144		2011-07-26	Drissi , et al.	
	30	8005828		2011-08-23	Buchner , et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	31	8019699		2011-09-13	Baxter	
	32	8078556		2011-12-13	Adi , et al.	
	33	8090669		2012-01-03	Shahani , et al.	
	34	8137112		2012-03-20	Woolf , et al.	
	35	8166463		2012-04-24	Gill , et al.	
	36	8195674		2012-06-05	Bem , et al.	
	37	8196119		2012-06-05	Gill , et al.	
	38	8244730		2012-08-14	Gupta	
	39	8251704		2012-08-28	Woolf , et al.	
	40	8266608		2012-09-11	Hecht , et al.	
	41	8364612		2013-01-29	Van Gael , et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	42	8386401		2013-02-26	Virkar , et al.	
	43	8397227		2013-03-12	Fan , et al.	
	44	8464225		2013-06-11	Greifeneder	
	45	8549359		2013-10-01	Zheng	
	46	8589414		2013-11-19	Waite , et al.	
	47	8655260		2014-02-18	Yueh , et al.	
	48	8667472		2014-03-04	Molinari	

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS Remove

Examiner Initial*	Cite No	Publication Number	Kind Code1	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	20120167057		2012-06-28	Schmich; Christopher P. ; et al.	
	2	20120150773		2012-06-14	DiCorpo; Phillip ; et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	3	20080254429		2008-10-16	Woolf; Susan D. ; et al.	
	4	20040117771		2004-06-17	Venkatapathy, Ramanathan	
	5	20130238533		2013-09-12	VIRKAR; Hemant ; et al.	
	6	20060190930		2006-08-24	Hecht; Daniel M. ; et al.	
	7	20070006159		2007-01-04	Hecht; Daniel Michael ; et al.	
	8	20080281764		2008-11-13	Baxter; Jonathan	
	9	20100138370		2010-06-03	Wu; Wang ; et al.	
	10	20110218672		2011-09-08	Maisonnier; Bruno ; et al.	

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number[3]	Country Code[2] i	Kind Code[4]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T[5]
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T5
	1	About Event Tracing, retrieved from <URL: http://msdn.microsoft.com/en-us/library/aa363668(d=default,l=en-us, v=vs.85).aspx> on Jan 12, 2014, 2 pages	
	2	Add dynamic Java code to your application, retrieved from <URL: http://www.javaworld.com/article/2071777/design-patterns/add-dynamic-java-code-to-your-application.html> on May 13, 2014, 6 pages	
	3	BCEL, retrieved from <URL: http://commons.apache.org> on May 13, 2014, 2 pages	
	4	Artificial intelligence, retrieved from <URL: http://wikipedia.com> on May 21, 2014, 28 pages	
	5	Artificial neural network, retrieved from <URL: http://wikipedia.com> on May 21, 2014, 12 pages	
	6	Branch (computer science), retrieved from <URL: http://wikipedia.com> on May 6, 2014, 2 pages	
	7	Branch trace, retrieved from <URL: http://wikipedia.com> on May 28, 2014, 2 pages	
	8	C Function Call Conventions and the Stack, retrieved from <URL: http://www.csee.umbc.edu/~chang/cs313.s02/stack.shtml> on Jan 11, 2014, 4 pages	
	9	Call stack, retrieved from <URL: http://wikipedia.com> on Jan 11, 2014, 5 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT
(Not for submission under 37 CFR 1.99)

Application Number	15382743
Filing Date	2016-12-19
First Named Inventor	Jasmin Cosic
Art Unit	
Examiner Name	
Attorney Docket Number	

	10	Program compilation and execution flow, retrieved from <URL: http://cs.stackexchange.com/questions/6187/program-compilation-and-execution-flow> on Jan 9, 2014, 2 pages	
	11	Control unit, retrieved from <URL: http://wikipedia.com> on May 25, 2014, 2 pages	
	12	Create dynamic applications with javax.tools, retrieved from <URL: http://www.ibm.com/developerworks/library/j-jcomp/> on May 13, 2014, 11 pages	
	13	Creating a Debugging and Profiling Agent with JVMTI, retrieved from <URL: http://www.oracle.com/technetwork/articles/java/jvmti-136367.html> on Jan 11, 2014, 18 pages	
	14	Decision tree learning, retrieved from <URL: http://wikipedia.com> on Jun 24, 2014, 5 pages	
	15	Dynamic Bytecode Instrumentation, retrieved from <URL: http://www.drdobbs.com/tools/dynamic-bytecode-instrumentation/184406433> on Apr 26, 2014, 6 pages	
	16	Dynamic programming language, retrieved from <URL: http://wikipedia.com> on May 7, 2014, 4 pages	
	17	Dynamic recompilation, retrieved from <URL: http://wikipedia.com> on May 19, 2014, 4 pages	
	18	eval(), retrieved from <URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/eval> on May 7, 2014, 13 pages	
	19	Function, retrieved from <URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Function> on May 7, 2014, 11 pages	
	20	FunctionEnter2 Function, retrieved from <URL: https://msdn.microsoft.com/en-us/library/aa964981(d=default,l=en-us,v=vs.110).aspx> on Mar 19, 2014, 2 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	21	FunctionLeave2 Function, retrieved from <URL: https://msdn.microsoft.com/en-us/library/aa964942(d=default,l=en-us, v=vs.110).aspx> on Mar 19, 2014, 1 pages	
	22	Heuristic, retrieved from <URL: http://wikipedia.com> on May 22, 2014, 5 pages	
	23	How to: Configure Trace Switches, retrieved from <URL: https://msdn.microsoft.com/en-us/library/vstudio/t06xyy08 (d=default,l=en-us,v=vs.100).aspx> on Mar 19, 2014, 3 pages	
	24	Indirect branch, retrieved from <URL: http://wikipedia.com> on May 5, 2014, 1 pages	
	25	Instruction cycle, retrieved from <URL: http://wikipedia.com> on Apr 30, 2014, 2 pages	
	26	Instruction set, retrieved from <URL: http://wikipedia.com> on Apr 29, 2014, 6 pages	
	27	Interrupt, retrieved from <URL: http://wikipedia.com> on May 15, 2014, 5 pages	
	28	Introduction to Instrumentation and Tracing, retrieved from <URL: http://msdn.microsoft.com/en-us/library/aa983649 (d=default,l=en-us,v=vs.71).aspx> on Jan 10, 2014, 2 pages	
	29	Java bytecode, retrieved from <URL: http://wikipedia.com> on May 27, 2014, 4 pages	
	30	Package java.util.logging, retrieved from <URL: http://docs.oracle.com/javase/1.5.0/docs/api/java/util/logging/package-summary.html> on Mar 19, 2014, 2 pages	
	31	javac - Java programming language compiler, retrieved from <URL: http://docs.oracle.com/javase/7/docs/technotes/tools/windows/javac.html> on May 13, 2014, 12 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	32	Interface JavaCompiler, retrieved from <URL: http://wikipedia.com> on May 13, 2014, 4 pages	
	33	Javassist, retrieved from <URL: http://wikipedia.com> on May 13, 2014, 2 pages	
	34	Just-in-time compilation, retrieved from <URL: http://wikipedia.com> on May 19, 2014, 5 pages	
	35	JVM Tool Interface, retrieved from <URL: http://docs.oracle.com/javase/7/docs/platform/jvmti/jvmti.html> on Apr 26, 2014, 127 pages	
	36	Machine code, retrieved from <URL: http://wikipedia.com> on Apr 30, 2014, 3 pages	
	37	Metaprogramming, retrieved from <URL: http://wikipedia.com> on May 8, 2014, 3 pages	
	38	Microsoft.VisualBasic.Logging Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/ms128122(d=default,l=en-us,v=vs.110).aspx> on Jan 13, 2014, 1 pages	
	39	Pin (computer program), retrieved from <URL: http://wikipedia.com> on May 6, 2014, 3 pages	
	40	Processor register, retrieved from <URL: http://wikipedia.com> on Apr 29, 2014, 4 pages	
	41	Profiling (computer programming), retrieved from <URL: http://wikipedia.com> on Jan 10, 2014, 4 pages	
	42	Profiling Overview, retrieved from <URL: http://msdn.microsoft.com/en-us/library/bb384493(d=default,l=en-us,v=vs.110).aspx> on Jan 12, 2014, 6 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	43	Program counter, retrieved from <URL: http://wikipedia.com> on Apr 24, 2014, 2 pages	
	44	ptrace, retrieved from <URL: http://wikipedia.com> on Apr 24, 2014, 2 pages	
	45	Reflection (computer programming), retrieved from <URL: http://wikipedia.com> on May 7, 2014, 5 pages	
	46	Class Runtime, retrieved from <URL: http://docs.oracle.com/javase/7/docs/api/java/lang/Runtime.html> on Jan 10, 2014, 10 pages	
	47	Self-modifying code, retrieved from <URL: http://wikipedia.com> on Jan 12, 2014, 6 pages	
	48	Subjective logic, retrieved from <URL: http://wikipedia.com> on May 22, 2014, 5 pages	
	49	System.CodeDom.Compiler Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/z6b99ydt(d=default,l=en-us,v=vs.110).aspx> on May 13, 2014, 2 pages	
	50	System.Reflection.Emit Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/xd5fw18y(d=default,l=en-us,v=vs.110).aspx> on May 13, 2014, 2 pages	

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature		Date Considered	

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Jasmin Cosic/	Date (YYYY-MM-DD)	2016-12-20
Name/Print	Jasmin Cosic	Registration Number	

This collection of information is required by 37 CFR 1.97 and 1.98. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 1 hour to complete, including gathering, preparing and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS

Remove

Examiner Initial*	Cite No	Patent Number	Kind Code[1]	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1					

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS

Remove

Examiner Initial*	Cite No	Publication Number	Kind Code[1]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1					

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS

Remove

Examiner Initial*	Cite No	Foreign Document Number[3]	Country Code[2] i	Kind Code[4]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T[5]
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS

Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T[5]

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	1	Tracing (software), retrieved from <URL: http://wikipedia.com> on Jan 10, 2014, 3 pages	
	2	Tree (data structure), retrieved from <URL: http://wikipedia.com> on Jun 24, 2014, 6 pages	
	3	PTRACE(2), retrieved from <URL: http://unixhelp.ed.ac.uk/CGI/man-cgi?ptrace> on Mar 19, 2014, 5 pages	
	4	Wevtutil, retrieved from <URL: http://technet.microsoft.com/en-us/library/cc732848(d=default,l=en-us,v=ws.11).aspx> on Apr 28, 2014, 5 pages	
	5	Intel Processor Trace, retrieved from <URL: https://software.intel.com/en-us/blogs/2013/09/18/processor-tracing> on Apr 28, 2014, 3 pages	
	6	YOUNGHOON JUNG, JAVA DYNAMICS Reflection and a lot more, Oct 10, 2012, 55 pages, Columbia University	
	7	AMITABH SRIVASTAVA, ALAN EUSTACE, ATOM A System for Building Customized Program Analysis Tools, May 3, 2004, 12 pages	
	8	MATHEW SMITHSON, KAPIL ANAND, APARNA KOTHA, KHALED ELWAZEER, NATHAN GILES, RAJEEV BARUA, Binary Rewriting without Relocation Information, Nov 10, 2010, 11 pages, University of Maryland	
	9	MAREK OLSZEWSKI, KEIR MIERTE, ADAM CZAJKOWSKI, ANGELA DEMLE BROWN, JIT Instrumentation - A Novel Approach To Dynamically Instrument Operating Systems, Feb 12, 2007, 14 pages, University of Toronto	

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature		Date Considered	

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Jasmin Cosic/	Date (YYYY-MM-DD)	2016-12-20
Name/Print	Jasmin Cosic	Registration Number	

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

PTO/SB/08a (01-10)
Approved for use through 07/31/2012. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS

Remove

Examiner Initial*	Cite No	Patent Number	Kind Code1	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1					

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS

Remove

Examiner Initial*	Cite No	Publication Number	Kind Code1	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	20030026588		2003-02-06	Elder, James H. ; et al.	
	2	20040249774		2004-12-09	Caid, William R. ; et al.	
	3	20040267521		2004-12-30	Cutler, Ross ; et al.	
	4	20050245303		2005-11-03	Graepel, Kurt Hartwig ; et al.	
	5	20060047612		2006-03-02	Stanley; Kenneth O. ; et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	6	20070058856		2007-03-15	Boregowda; Lokesh R. ; et al.	
	7	20070106633		2007-05-10	Reiner; Bruce	
	8	20080144893		2008-06-19	GUO; Chun Biao ; et al.	
	9	20080288259		2008-11-20	Chambers; Robert L. ; et al.	
	10	20090110061		2009-04-30	LEE; Kang Yong ; et al.	
	11	20090131152		2009-05-21	Busse; Martin A.	
	12	20090141969		2009-06-04	Yu; Kai ; et al.	
	13	20090222388		2009-09-30	Hua; Wei ; et al.	
	14	20090324010		2009-12-31	Hou; Billy	
	15	20100033780		2010-02-11	Gitter; Kurt	
	16	20100063949		2010-03-11	EATON; JOHN ERIC ; et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	17	20100241595		2010-09-23	Felsher; David Paul	
	18	20100278420		2010-11-04	Shet; Vinay Damodar ; et al.	
	19	20110030031		2011-02-03	Lussier; Paul ; et al.	
	20	20130156345		2013-06-20	Shmunk; Dmitry Valerievich	
	21	20130159021		2013-06-20	Felsher; David Paul	
	22	20130278631		2013-10-24	Border; John N. ; et al.	
	23	20140052717		2014-02-20	FRANKS; Kasian ; et al.	
	24	20140161250		2014-06-12	Leeds; Richard ; et al.	
	25	20140177946		2014-06-26	LIM; Kil-Taek ; et al.	
	26	20140207580		2014-06-24	Minnis; William Charles ; et al.	
	27	20150039304		2015-02-05	Wein; Ron	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	28	20150055821		2015-02-26	Fotland; David Allen

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number3	Country Code2 i	Kind Code4	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T5
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T5
	1	ABOUT | OpenCV, retrieved from <URL: http://opencv.org/about.html> on Dec 13, 2014, 1 pages	
	2	Animetrics, Inc. | 3D Facial Recognition, retrieved from <URL: http://animetrics.com/> on Dec 13, 2014, 2 pages	
	3	Bag-of-words model, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	4	Beier–Neely morphing algorithm, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 1 pages	
	5	Computer vision, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 7 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	6	Convolutional neural network, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 5 pages	
	7	Decimation (signal processing), retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 3 pages	
	8	Digital image processing, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 3 pages	
	9	Dissolve (filmmaking), retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 2 pages	
	10	Facial recognition system, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 6 pages	
	11	Feature detection (computer vision), retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 3 pages	
	12	Feature extraction, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	13	Gesture recognition, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 5 pages	
	14	GrabCut, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 1 pages	
	15	Image processing, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	16	Image segmentation, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 12 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	17	Image warping, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 2 pages	
	18	Inbetweening, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 2 pages	
	19	Interpolation, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 5 pages	
	20	Language model, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 4 pages	
	21	List of speech recognition software, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 4 pages	
	22	Livewire Segmentation Technique, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 2 pages	
	23	Morphing, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 2 pages	
	24	Motion estimation, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	25	Muse: Face Morph || Mesh Warping, retrieved from <URL: http://alexwolfe.blogspot.com/2011/10/face-morph-mesh-warping.html> on Nov 19, 2015, 2 pages	
	26	n-gram, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 5 pages	
	27	Optical flow, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 4 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	28	Outline of object recognition, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 7 pages	
	29	Phonotactics, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 3 pages	
	30	Recurrent neural network, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 5 pages	
	31	Sample rate conversion, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 2 pages	
	32	Simple interactive object extraction, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	33	Speech recognition, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 12 pages	
	34	Speech segmentation, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 4 pages	
	35	Welcome to recognize-speech.com, retrieved from <URL: http://recognize-speech.com/> on Oct 18, 2015, 1 pages	
	36	Introduction Speech, retrieved from <URL: http://recognize-speech.com/speech> on Oct 18, 2015, 1 pages	
	37	Preprocessing, retrieved from <URL: http://recognize-speech.com/preprocessing> on Oct 18, 2015, 4 pages	
	38	Feature Extraction, retrieved from <URL: http://recognize-speech.com/feature-extraction> on Oct 18, 2015, 3 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	39	Acoustic model, retrieved from <URL: http://recognize-speech.com/acoustic-model> on Oct 18, 2015, 2 pages	
	40	Video content analysis, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	41	Video tracking, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 2 pages	
	42	Andrej Karpathy, Thomas Leung, George Toderici, Rahul Sukthankar, Sanketh Shetty, Li Fei-Fei, Large-scale Video Classification with Convolutional Neural Networks, Apr 14, 2014, 8 pages, Stanford University	
	43	Karen Simonyan, Andrew Zisserman, Two-Stream Convolutional Networks for Action Recognition in Videos, Nov 13, 2014, 11 pages, University of Oxford	

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature		Date Considered	

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number		15382743
	Filing Date		2016-12-19
	First Named Inventor	Jasmin Cosic	
	Art Unit		
	Examiner Name		
	Attorney Docket Number		

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Jasmin Cosic/	Date (YYYY-MM-DD)	2016-12-20
Name/Print	Jasmin Cosic	Registration Number	

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed
PTO/SB/08a (01-10)
Approved for use through 07/31/2012. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS Remove

Examiner Initial*	Cite No	Patent Number	Kind Code1	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	5758333		1998-05-26	Bauer , et al.	
	2	7409401		2008-08-05	Hansen , et al.	
	3	7533128		2009-05-12	Sanchez , et al.	
	4	7831564		2010-11-09	Wei , et al.	
	5	7444338		2008-10-28	Fisher	
	6	7082435		2006-07-25	Guzman , et al.	
	7	8949186		2015-02-03	Yueh , et al.	
	8	8762428		2014-06-24	Kulack , et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	9	7395255		2008-07-01	Li	
	10	6850942		2005-02-01	Cotner , et al.	
	11	5983232		1999-11-09	Zhang	
	12	5592661		1997-01-07	Eisenberg , et al.	
	13	7849114		2010-12-07	Boss , et al.	

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS Remove

Examiner Initial*	Cite No	Publication Number	Kind Code[1]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	20120089570		2012-04-12	Zha; Charles Li ; et al.	
	2	20140143276		2014-05-22	ROGERS; Michael Anthony ; et al.	
	3	20080071770		2008-03-20	Schloter; Philipp ; et al.	
	4	20110004586		2011-01-06	Cherryholmes; Lon Jones ; et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	5	20050154745		2005-07-14	Hansen, Lynda A. ; et al.	
	6	20140164430		2014-06-12	Hadjieleftheriou; Marios ; et al.	
	7	20020019822		2002-02-14	Seki, Yumiko ; et al.	
	8	20130204907		2013-08-08	Alonso Alarcon; Antonio ; et al.	
	9	20060259466		2006-11-16	Bilotti; David	
	10	20110093435		2011-04-21	Zha; Charlie Li ; et al.	
	11	20090070337		2009-03-12	Romem; Yaniv ; et al.	

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number³	Country Code² i	Kind Code⁴	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T⁵
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS Remove

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T5
	1	2D computer graphics, retrieved from <URL: http://wikipedia.com> on Nov 9, 2014, 7 pages	
	2	3D computer graphics, retrieved from <URL: http://wikipedia.com> on Nov 9, 2014, 3 pages	
	3	3D modeling, retrieved from <URL: http://wikipedia.com> on Nov 9, 2014, 5 pages	
	4	8: Mouse Handling, retrieved from <URL: http://ericsink.com/wpf3d/8_Mouse.html> on Nov 13, 2014, 2 pages	
	5	AutoCAD, retrieved from <URL: http://wikipedia.com> on Nov 9, 2014, 7 pages	
	6	Collision detection, retrieved from <URL: http://wikipedia.com> on Nov 23, 2014, 5 pages	
	7	Comparison of OpenGL and Direct3D, retrieved from <URL: http://wikipedia.com> on Nov 22, 2014, 7 pages	
	8	Data dictionary, retrieved from <URL: http://wikipedia.com> on Nov 6, 2014, 3 pages	
	9	DBA_TABLES, retrieved from <URL: http://wikipedia.com> on Nov 6, 2014, 1 pages	
	10	Digital sculpting, retrieved from <URL: http://wikipedia.com> on Nov 12, 2014, 3 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	11	Facial recognition system, retrieved from <URL: http://wikipedia.com> on Dec 13, 2014, 7 pages	
	12	Game engine, retrieved from <URL: http://wikipedia.com> on Nov 23, 2014, 4 pages	
	13	JavaFX, retrieved from <URL: http://wikipedia.com> on Nov 22, 2014, 7 pages	
	14	Ureality, retrieved from <URL: http://wikipedia.com> on Nov 22, 2014, 3 pages	
	15	List of 14+ Image Recognition APIs, retrieved from <URL: http://blog.mashape.com/list-of-14-image-recognition-apis/> on Dec 13, 2014, 2 pages	
	16	List of 50+ Face Detection / Recognition APIs, libraries, and software, retrieved from <URL: http://blog.mashape.com/list-of-50-face-detection-recognition-apis/> on Dec 13, 2014, 4 pages	
	17	List of file formats, retrieved from <URL: http://wikipedia.com> on Nov 9, 2014, 33 pages	
	18	List of game engines, retrieved from <URL: http://wikipedia.com> on Nov 22, 2014, 17 pages	
	19	Lua (programming language), retrieved from <URL: http://wikipedia.com> on Nov 25, 2014, 10 pages	
	20	Mantle (API), retrieved from <URL: http://wikipedia.com> on Nov 22, 2014, 5 pages	
	21	Microsoft XNA, retrieved from <URL: http://wikipedia.com> on Nov 22, 2014, 7 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number		15382743
	Filing Date		2016-12-19
	First Named Inventor	Jasmin Cosic	
	Art Unit		
	Examiner Name		
	Attorney Docket Number		

	22	3.4 Getting Information About Databases and Tables, retrieved from <URL: http://dev.mysql.com/doc/refman/5.7/en/getting-information.html> on Nov 6, 2014, 2 pages	
	23	MySQL show users - how to show the users in a MySQL database, retrieved from <URL: http://alvinalexander.com/blog/post/mysql/show-users-i-ve-created-in-mysql-database> on Nov 6, 2014, 4 pages	
	24	OpenCV, retrieved from <URL: http://wikipedia.com> on Dec 13, 2014, 3 pages	
	25	Optical character recognition, retrieved from <URL: http://wikipedia.com> on Dec 13, 2014, 6 pages	
	26	Outline of object recognition, retrieved from <URL: http://wikipedia.com> on Dec 13, 2014, 8 pages	
	27	Particle system, retrieved from <URL: http://wikipedia.com> on Nov 12, 2014, 3 pages	
	28	Physics engine, retrieved from <URL: http://wikipedia.com> on Nov 23, 2014, 5 pages	
	29	Polygonal modeling, retrieved from <URL: http://wikipedia.com> on Nov 12, 2014, 4 pages	
	30	POV-Ray, retrieved from <URL: http://wikipedia.com> on Nov 10, 2014, 5 pages	
	31	Raster graphics, retrieved from <URL: http://wikipedia.com> on Nov 9, 2014, 3 pages	
	32	Scene graph, retrieved from <URL: http://wikipedia.com> on Nov 22, 2014, 4 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	33	STL (file format), retrieved from <URL: http://wikipedia.com> on Nov 11, 2014, 4 pages	
	34	Subdivision surface, retrieved from <URL: http://wikipedia.com> on Nov 12, 2014, 3 pages	
	35	sysobjects, retrieved from <URL: http://technet.microsoft.com/en-us/library/aa260447(d=default,l=en-us,v=sql.80).aspx> on Nov 6, 2014, 2 pages	
	36	Vector graphics, retrieved from <URL: http://wikipedia.com> on Nov 9, 2014, 5 pages	
	37	Vertex (computer graphics), retrieved from <URL: http://wikipedia.com> on Nov 9, 2014, 2 pages	
	38	Voxel, retrieved from <URL: http://wikipedia.com> on Nov 11, 2014, 5 pages	
	39	Wavefront .obj file, retrieved from <URL: http://wikipedia.com> on Nov 9, 2014, 6 pages	
	40	Acrobat 3D tutorials - basic interaction, retrieved from <URL: https://acrobatusers.com/assets/collections/tutorials/legacy/tech_corners/3d/3d_tutorials/basic_interaction.pdf> on Nov 23, 2014, 2 pages	

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature		Date Considered	

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Jasmin Cosic/	Date (YYYY-MM-DD)	2016-12-20
Name/Print	Jasmin Cosic	Registration Number	

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

PTO/SB/08a (01-10)
Approved for use through 07/31/2012. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS Remove

Examiner Initial*	Cite No	Patent Number	Kind Code1	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	7113946		2006-09-26	Cosic	
	2	7117225		2006-10-03	Cosic	
	3	8335805		2012-12-18	Cosic	
	4	8417740		2013-04-09	Cosic	
	5	8572035		2013-10-29	Cosic	
	6	8655900		2014-02-18	Cosic	
	7	9047324		2015-06-02	Cosic	
	8	9282309		2016-03-08	Cosic	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	9	9298749		2016-03-29	Cosic	
	10	9367806		2016-06-14	Cosic	
	11	9443192		2016-09-13	Cosic	

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS Remove

Examiner Initial*	Cite No	Publication Number	Kind Code[1]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	20030065662		2003-04-03	Cosic	
	2	20040194017		2004-09-30	Cosic	
	3	20050149517		2005-07-07	Cosic	
	4	20050149542		2005-07-07	Cosic	
	5	20050289105		2005-12-29	Cosic	
	6	20100023541		2010-01-28	Cosic	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	7	20100082536		2010-04-01	Cosic	
	8	20130218932		2013-08-22	Cosic	
	9	20130226974		2013-08-29	Cosic	
	10	20160140999		2016-05-19	Cosic	
	11	20160142650		2016-05-19	Cosic	
	12	20160246819		2016-08-25	Cosic	
	13	20160246850		2016-08-25	Cosic	
	14	20160246868		2016-08-25	Cosic	
	15	20160292185		2016-10-06	Cosic	

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number[3]	Country Code[2] i	Kind Code[4]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T[5]

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T[5]
	1		

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature		Date Considered	

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

X A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Jasmin Cosic/	Date (YYYY-MM-DD)	2016-12-20
Name/Print	Jasmin Cosic	Registration Number	

This collection of information is required by 37 CFR 1.97 and 1.98. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 1 hour to complete, including gathering, preparing and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Electronic Acknowledgement Receipt

EFS ID:	27841167
Application Number:	15382743
International Application Number:	
Confirmation Number:	4742
Title of Invention:	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING AN AVATAR'S CIRCUMSTANCES FOR AUTONOMOUS AVATAR OPERATION
First Named Inventor/Applicant Name:	Jasmin Cosic
Customer Number:	116094
Filer:	Jasmin Cosic
Filer Authorized By:	
Attorney Docket Number:	
Receipt Date:	20-DEC-2016
Filing Date:	
Time Stamp:	03:03:39
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	no

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1	Information Disclosure Statement (IDS) Form (SB08)	IDS_Part1.pdf	615768 5a224e61e706cf3b86fda507afbfedd3ce4a6f3a	no	13

Warnings:

Information:

2	Information Disclosure Statement (IDS) Form (SB08)	IDS_Part2.pdf	613084 6ff1c67121359915e31ab3f5fe637ee63bd41e05	no	5

Warnings:

Information:

A U.S. Patent Number Citation or a U.S. Publication Number Citation is required in the Information Disclosure Statement (IDS) form for autoloading of data into USPTO systems. You may remove the form to add the required data in order to correct the Informational Message if you are citing U.S. References. If you chose not to include U.S. References, the image of the form will be processed and be made available within the Image File Wrapper (IFW) system. However, no data will be extracted from this form. Any additional data such as Foreign Patent Documents or Non Patent Literature will be manually reviewed and keyed into USPTO systems.

3	Information Disclosure Statement (IDS) Form (SB08)	IDS_Part3.pdf	614633 67c8b2c47c969c7a3ecf9e0154502e38a25d7647	no	10

Warnings:

Information:

4	Information Disclosure Statement (IDS) Form (SB08)	IDS_Part4.pdf	614432 830d6203b5a5fe268f6c6571eb17efec6eedc921	no	9

Warnings:

Information:

5	Information Disclosure Statement (IDS) Form (SB08)	IDS_Part5.pdf	612877 2cebcd51e6a0183d17d0cb4b677c1f455405e3f0	no	6

Warnings:

Information:

6	Non Patent Literature	NPL_Part1.pdf	19779115 ee07479784f30836d39a22bda6b5ec735694f56e	no	380

Warnings:

The page size in the PDF is too large. The pages should be 8.5 x 11 or A4. If this PDF is submitted, the pages will be resized upon entry into the Image File Wrapper and may affect subsequent processing

Information:

7	Non Patent Literature	NPL_Part2.pdf	5879608 8fd00edc18b82bf74a8ebf692fa5e39b231f3d56	no	114

Warnings:

The page size in the PDF is too large. The pages should be 8.5 x 11 or A4. If this PDF is submitted, the pages will be resized upon entry into the Image File Wrapper and may affect subsequent processing

Information:

8	Non Patent Literature	NPL_Part3.pdf	16903333 327e784f4ab311324ac99b3c6dd8b06ae0b170c7	no	160
Warnings:					
Information:					
9	Non Patent Literature	NPL_Part4.pdf	23664072 9f7da794ce4e397cebac905b658a35e81e7d7f4c	no	218
Warnings:					
Information:					
			Total Files Size (in bytes):	69296922	

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

<u>New Applications Under 35 U.S.C. 111</u>
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.

<u>National Stage of an International Application under 35 U.S.C. 371</u>
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

<u>New International Application Filed with the USPTO as a Receiving Office</u>
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NUMBER	FILING or 371(c) DATE	GRP ART UNIT	FIL FEE REC'D	ATTY.DOCKET.NO	TOT CLAIMS	IND CLAIMS
15/382,743	12/19/2016	2129	930		20	3

CONFIRMATION NO. 4742

FILING RECEIPT

OC000000088129630

116094
Jasmin Cosic
108 Woodbury Street
Pawtucket, RI 02861

Date Mailed: 12/29/2016

Receipt is acknowledged of this non-provisional patent application. The application will be taken up for examination in due course. Applicant will be notified as to the results of the examination. Any correspondence concerning the application must include the following identification information: the U.S. APPLICATION NUMBER, FILING DATE, NAME OF APPLICANT, and TITLE OF INVENTION. Fees transmitted by check or draft are subject to collection. Please verify the accuracy of the data presented on this receipt. **If an error is noted on this Filing Receipt, please submit a written request for a Filing Receipt Correction. Please provide a copy of this Filing Receipt with the changes noted thereon. If you received a "Notice to File Missing Parts" for this application, please submit any corrections to this Filing Receipt with your reply to the Notice. When the USPTO processes the reply to the Notice, the USPTO will generate another Filing Receipt incorporating the requested corrections**

Inventor(s)

Jasmin Cosic, Miami, FL;

Applicant(s)

Jasmin Cosic, Miami, FL;

Power of Attorney: None

Domestic Applications for which benefit is claimed - None.
A proper domestic benefit claim must be provided in an Application Data Sheet in order to constitute a claim for domestic benefit. See 37 CFR 1.76 and 1.78.

Foreign Applications for which priority is claimed (You may be eligible to benefit from the **Patent Prosecution Highway** program at the USPTO. Please see http://www.uspto.gov for more information.) - None.
Foreign application information must be provided in an Application Data Sheet in order to constitute a claim to foreign priority. See 37 CFR 1.55 and 1.76.

Permission to Access Application via Priority Document Exchange: No

Permission to Access Search Results: No

Applicant may provide or rescind an authorization for access using Form PTO/SB/39 or Form PTO/SB/69 as appropriate.

If Required, Foreign Filing License Granted: 12/27/2016

The country code and number of your priority application, to be used for filing abroad under the Paris Convention, is **US 15/382,743**

Projected Publication Date: Request for Non-Publication Acknowledged

Non-Publication Request: Yes

Early Publication Request: No
**** SMALL ENTITY ****
Title

ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING AN AVATAR'S CIRCUMSTANCES FOR AUTONOMOUS AVATAR OPERATION

Preliminary Class

706

Statement under 37 CFR 1.55 or 1.78 for AIA (First Inventor to File) Transition Applications: No

PROTECTING YOUR INVENTION OUTSIDE THE UNITED STATES

Since the rights granted by a U.S. patent extend only throughout the territory of the United States and have no effect in a foreign country, an inventor who wishes patent protection in another country must apply for a patent in a specific country or in regional patent offices. Applicants may wish to consider the filing of an international application under the Patent Cooperation Treaty (PCT). An international (PCT) application generally has the same effect as a regular national patent application in each PCT-member country. The PCT process **simplifies** the filing of patent applications on the same invention in member countries, but **does not result** in a grant of "an international patent" and does not eliminate the need of applicants to file additional documents and fees in countries where patent protection is desired.

Almost every country has its own patent law, and a person desiring a patent in a particular country must make an application for patent in that country in accordance with its particular laws. Since the laws of many countries differ in various respects from the patent law of the United States, applicants are advised to seek guidance from specific foreign countries to ensure that patent rights are not lost prematurely.

Applicants also are advised that in the case of inventions made in the United States, the Director of the USPTO must issue a license before applicants can apply for a patent in a foreign country. The filing of a U.S. patent application serves as a request for a foreign filing license. The application's filing receipt contains further information and guidance as to the status of applicant's license for foreign filing.

Applicants may wish to consult the USPTO booklet, "General Information Concerning Patents" (specifically, the section entitled "Treaties and Foreign Patents") for more information on timeframes and deadlines for filing foreign patent applications. The guide is available either by contacting the USPTO Contact Center at 800-786-9199, or it can be viewed on the USPTO website at http://www.uspto.gov/web/offices/pac/doc/general/index.html.

For information on preventing theft of your intellectual property (patents, trademarks and copyrights), you may wish to consult the U.S. Government website, http://www.stopfakes.gov. Part of a Department of Commerce initiative, this website includes self-help "toolkits" giving innovators guidance on how to protect intellectual property in specific countries such as China, Korea and Mexico. For questions regarding patent enforcement issues, applicants may call the U.S. Government hotline at 1-866-999-HALT (1-866-999-4258).

LICENSE FOR FOREIGN FILING UNDER

Title 35, United States Code, Section 184

Title 37, Code of Federal Regulations, 5.11 & 5.15

GRANTED

The applicant has been granted a license under 35 U.S.C. 184, if the phrase "IF REQUIRED, FOREIGN FILING LICENSE GRANTED" followed by a date appears on this form. Such licenses are issued in all applications where the conditions for issuance of a license have been met, regardless of whether or not a license may be required as set forth in 37 CFR 5.15. The scope and limitations of this license are set forth in 37 CFR 5.15(a) unless an earlier license has been issued under 37 CFR 5.15(b). The license is subject to revocation upon written notification. The date indicated is the effective date of the license, unless an earlier license of similar scope has been granted under 37 CFR 5.13 or 5.14.

This license is to be retained by the licensee and may be used at any time on or after the effective date thereof unless it is revoked. This license is automatically transferred to any related applications(s) filed under 37 CFR 1.53(d). This license is not retroactive.

The grant of a license does not in any way lessen the responsibility of a licensee for the security of the subject matter as imposed by any Government contract or the provisions of existing laws relating to espionage and the national security or the export of technical data. Licensees should apprise themselves of current regulations especially with respect to certain countries, of other agencies, particularly the Office of Defense Trade Controls, Department of State (with respect to Arms, Munitions and Implements of War (22 CFR 121-128)); the Bureau of Industry and Security, Department of Commerce (15 CFR parts 730-774); the Office of Foreign AssetsControl, Department of Treasury (31 CFR Parts 500+) and the Department of Energy.

NOT GRANTED

No license under 35 U.S.C. 184 has been granted at this time, if the phrase "IF REQUIRED, FOREIGN FILING LICENSE GRANTED" DOES NOT appear on this form. Applicant may still petition for a license under 37 CFR 5.12, if a license is desired before the expiration of 6 months from the filing date of the application. If 6 months has lapsed from the filing date of this application and the licensee has not received any indication of a secrecy order under 35 U.S.C. 181, the licensee may foreign file the application pursuant to 37 CFR 5.15(b).

PATENT APPLICATION FEE DETERMINATION RECORD

Substitute for Form PTO-875

Application or Docket Number
15/382,743

APPLICATION AS FILED - PART I

FOR	NUMBER FILED (Column 1)	NUMBER EXTRA (Column 2)	SMALL ENTITY RATE($)	SMALL ENTITY FEE($)	OR	OTHER THAN SMALL ENTITY RATE($)	OTHER THAN SMALL ENTITY FEE($)
BASIC FEE (37 CFR 1.16(a), (b), or (c))	N/A	N/A	N/A	70		N/A	
SEARCH FEE (37 CFR 1.16(k), (i), or (m))	N/A	N/A	N/A	300		N/A	
EXAMINATION FEE (37 CFR 1.16(o), (p), or (q))	N/A	N/A	N/A	360		N/A	
TOTAL CLAIMS (37 CFR 1.16(i))	20 minus 20 =	*	x 40 =	0.00	OR		
INDEPENDENT CLAIMS (37 CFR 1.16(h))	3 minus 3 =	*	x 210 =	0.00			
APPLICATION SIZE FEE (37 CFR 1.16(s))	If the specification and drawings exceed 100 sheets of paper, the application size fee due is $310 ($155 for small entity) for each additional 50 sheets or fraction thereof. See 35 U.S.C. 41(a)(1)(G) and 37 CFR 1.16(s).			200			
MULTIPLE DEPENDENT CLAIM PRESENT (37 CFR 1.16(j))				0.00			
* If the difference in column 1 is less than zero, enter "0" in column 2.			TOTAL	930		TOTAL	

APPLICATION AS AMENDED - PART II

AMENDMENT A

	CLAIMS REMAINING AFTER AMENDMENT (Column 1)		HIGHEST NUMBER PREVIOUSLY PAID FOR (Column 2)	PRESENT EXTRA (Column 3)	SMALL ENTITY RATE($)	SMALL ENTITY ADDITIONAL FEE($)	OR	OTHER THAN SMALL ENTITY RATE($)	OTHER THAN SMALL ENTITY ADDITIONAL FEE($)
Total (37 CFR 1.16(i))	*	Minus	**	=	x =		OR	x =	
Independent (37 CFR 1.16(h))	*	Minus	***	=	x =		OR	x =	
Application Size Fee (37 CFR 1.16(s))									
FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))							OR		
					TOTAL ADD'L FEE		OR	TOTAL ADD'L FEE	

AMENDMENT B

	CLAIMS REMAINING AFTER AMENDMENT (Column 1)		HIGHEST NUMBER PREVIOUSLY PAID FOR (Column 2)	PRESENT EXTRA (Column 3)	RATE($)	ADDITIONAL FEE($)		RATE($)	ADDITIONAL FEE($)
Total (37 CFR 1.16(i))	*	Minus	**	=	x =		OR	x =	
Independent (37 CFR 1.16(h))	*	Minus	***	=	x =		OR	x =	
Application Size Fee (37 CFR 1.16(s))									
FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))							OR		
					TOTAL ADD'L FEE		OR	TOTAL ADD'L FEE	

* If the entry in column 1 is less than the entry in column 2, write "0" in column 3.
** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, enter "20".
*** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, enter "3".
The "Highest Number Previously Paid For" (Total or Independent) is the highest found in the appropriate box in column 1.

PTO/SB/06 (09-11)
Approved for use through 1/31/2014. OMB 0651-0032
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

PATENT APPLICATION FEE DETERMINATION RECORD Substitute for Form PTO-875	Application or Docket Number 15/382,743	Filing Date 12/19/2016	☐ To be Mailed

ENTITY: ☐ LARGE ☑ SMALL ☐ MICRO

APPLICATION AS FILED - PART I

	(Column 1)	(Column 2)			
FOR	NUMBER FILED	NUMBER EXTRA		RATE ($)	FEE ($)
☐ BASIC FEE (37 CFR 1.16(a), (b), or (c))	N/A	N/A		N/A	
☐ SEARCH FEE (37 CFR 1.16(k), (i), or (m))	N/A	N/A		N/A	
☐ EXAMINATION FEE (37 CFR 1.16(o), (p), or (q))	N/A	N/A		N/A	
TOTAL CLAIMS (37 CFR 1.16(i))	minus 20 =	*		x $40 =	
INDEPENDENT CLAIMS (37 CFR 1.16(h))	minus 3 =	*		x $210 =	
☐ APPLICATION SIZE FEE (37 CFR 1.16(s))	If the specification and drawings exceed 100 sheets of paper, the application size fee due is $310 ($155 for small entity) for each additional 50 sheets or fraction thereof. See 35 U.S.C. 41(a)(1)(G) and 37 CFR 1.16(s).				
☐ MULTIPLE DEPENDENT CLAIM PRESENT (37 CFR 1.16(j))					
* If the difference in column 1 is less than zero, enter "0" in column 2.				TOTAL	

APPLICATION AS AMENDED - PART II

		(Column 1)		(Column 2)	(Column 3)			
AMENDMENT	04/25/2019	CLAIMS REMAINING AFTER AMENDMENT		HIGHEST NUMBER PREVIOUSLY PAID FOR	PRESENT EXTRA		RATE ($)	ADDITIONAL FEE ($)
	Total (37 CFR 1.16(i))	* 20	Minus	** 20	= 0		x $50 =	0
	Independent (37 CFR 1.16(h))	* 3	Minus	*** 3	= 0		x $230 =	0
	☐ Application Size Fee (37 CFR 1.16(s))							
	☐ FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))							
							TOTAL ADD'L FEE	0

		(Column 1)		(Column 2)	(Column 3)			
AMENDMENT		CLAIMS REMAINING AFTER AMENDMENT		HIGHEST NUMBER PREVIOUSLY PAID FOR	PRESENT EXTRA		RATE ($)	ADDITIONAL FEE ($)
	Total (37 CFR 1.16(i))	*	Minus	**	=		x $0 =	
	Independent (37 CFR 1.16(h))	*	Minus	***	=		x $0 =	
	☐ Application Size Fee (37 CFR 1.16(s))							
	☐ FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))							
							TOTAL ADD'L FEE	

* If the entry in column 1 is less than the entry in column 2, write "0" in column 3.	SLIE
** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, enter "20".	/SHARON M WEST/
*** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, enter "3".	
The "Highest Number Previously Paid For" (Total or Independent) is the highest number found in the appropriate box in column 1.	

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Inventor:	Jasmin Cosic	Confirmation No.: 4742
Title:	"ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING AN AVATAR'S CIRCUMSTANCES FOR AUTONOMOUS AVATAR OPERATION"	
Serial No.:	15/382,743	Filed: December 20, 2016
Examiner:	CHAKI, KAKALI	Group Art Unit: 2122

Via EFS-Web April 25, 2019

Mail Stop Amendment
COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, VA 22313-1450

PRELIMINARY AMENDMENT

Dear Commissioner:

Before examination on the merits, please amend the claims in the above-identified application as shown in the following listing of claims.

Listing of Claims

1 - 20 (canceled)

21. (new) A system comprising:

one or more processor circuits configured to execute instruction sets of an application;

a memory that stores at least a first correlation including a first one or more object representations correlated with a first one or more instruction sets for operating a first avatar of the application and a second correlation including a second one or more object representations correlated with a second one or more instruction sets for operating the first avatar of the application, wherein the first one or more object representations represent a first one or more objects of the application and the second one or more object representations represent a second one or more objects of the application, and wherein at least a portion of the first correlation and at least a portion of the second correlation are learned in a learning process while the first avatar of the application is at least partially operated by a user; and

an artificial intelligence unit that:

generates or receives a third one or more object representations, wherein the third one or more object representations represent a third one or more objects of the application;

determines the first one or more instruction sets for operating the first avatar of the application based on at least partial match between the third one or more object representations and the first one or more object representations; and

in response to the determines of the artificial intelligence unit, causes the first avatar of the application or a second avatar of the application to autonomously perform one or more operations defined by the first one or more instruction sets for operating the first avatar of the application at least by causing the one or more processor circuits to execute the first one or more instruction sets for operating the first avatar of the application.

22. (new) The system of claim 21, wherein the first one or more object representations include a first stream of one or more object representations, and wherein the second one or more object representations include a second stream of one or more object representations, and wherein the third one or more object representations include a third stream of one or more object representations.

23. (new) The system of claim 21, wherein the memory further stores at least a first knowledge cell and a second knowledge cell, and wherein the first knowledge cell includes the first correlation and the second knowledge cell includes the second correlation.

24. (new) The system of claim 21, wherein the learning process includes:

generating or receiving the first one or more object representations;

obtaining or receiving the first one or more instruction sets for operating the first avatar of the application;

generating or receiving the second one or more object representations; and

obtaining or receiving the second one or more instruction sets for operating the first avatar of the application.

25. (new) The system of claim 21, wherein the determines the first one or more instruction sets for operating the first avatar of the application based on the at least partial match between the third one or more object representations and the first one or more object representations includes:

determining that a number of at least partially matching portions of the third one or more object representations and portions of the first one or more object representations exceeds a threshold number; or

determining that a percentage of at least partially matching portions of the third one or more object representations and portions of the first one or more object representations exceeds a threshold percentage.

26. (new) The system of claim 21, wherein, to generate the first correlation, a determination is made that the first one or more instruction sets for operating the first avatar of the application temporally correspond to the first one or more object representations, and wherein, to generate the second correlation, a determination is made that the second one or more instruction sets for operating

the first avatar of the application temporally correspond to the second one or more object representations.

27. (new) The system of claim 21, wherein at least one object of the application of the first one or more objects of the application and at least one object of the application of the third one or more objects of the application are the same.

28. (new) The system of claim 21, wherein the artificial intelligence unit includes at least one selected from the group consisting of: a hardware element that is included in the one or more processor circuits, a hardware element that is included in another one or more processor circuits, a program operating on the one or more processor circuits, a program operating on another one or more processor circuits, and an element coupled to the one or more processor circuits, and wherein the application includes: a video game, a computer game, a simulation program, a program including text processing, a program including number processing, a program including picture processing, a program including object processing, or a program, and wherein the application includes one or more versions of the application, one or more upgrades of the application, one or more sequels of the application, one or more instances of the application, or one or more variations of the application, and wherein the learning process includes one or more learning processes, and wherein the first avatar of the application is a first object of the application and the second avatar of the application is a second object of the application.

29. (new) The system of claim 21, wherein the artificial intelligence unit includes at least one selected from the group consisting of: at least a portion of an object processing unit, at least a portion of an acquisition interface, at least a portion of a modification interface, and at least a portion of an ACAAO unit.

30. (new) The system of claim 21, wherein the first one or more objects of the application include one or more objects of the application in the first avatar's surrounding, and wherein the third one or more objects of the application include one or more objects of the application in the first avatar's surrounding, and wherein the first one or more instruction sets for operating the first avatar of the application are applied to the first avatar of the application so that the first avatar of the application autonomously performs the one or more operations defined by the first one or more instruction sets for operating the first avatar of the application.

31. (new) The system of claim 21, wherein the first one or more objects of the application include one or more objects of the application in the first avatar's surrounding, and wherein the third one or more objects of the application include one or more objects of the application in the second avatar's surrounding, and wherein the first one or more instruction sets for operating the first avatar of the application are applied to the second avatar of the application so that the second avatar of the application autonomously performs the one or more operations

defined by the first one or more instruction sets for operating the first avatar of the application.

32. (new) The system of claim 31, wherein the first avatar's surrounding includes at least one selected from the group consisting of:

a part of the application in an area of interest around the first avatar,

a part of the application defined by a threshold distance from the first avatar,

a part of the application relative to the first avatar,

a part of the application around the first avatar,

a part of the application that is shown to the user, and

at least a part of the application, and wherein the second avatar's surrounding includes at least one selected from the group consisting of:

a part of the application in an area of interest around the second avatar,

a part of the application defined by a threshold distance from the second avatar,

a part of the application relative to the second avatar,

a part of the application around the second avatar,

a part of the application that is shown to the user, and

at least a part of the application.

33. (new) The system of claim 21, wherein the first one or more objects of the application include one or more objects of the application in the first avatar's

surrounding, and wherein the third one or more objects of the application include one or more objects of the application in the second avatar's surrounding, and wherein the first one or more instruction sets for operating the first avatar of the application are modified and applied to the second avatar of the application so that the second avatar of the application autonomously performs the one or more operations defined by the modified first one or more instruction sets for operating the first avatar of the application.

34. (new) The system of claim 21, wherein the memory further stores at least a fourth correlation including a fourth one or more object representations correlated with a fourth one or more instruction sets for operating the first avatar of the application, and wherein the fourth one or more object representations represent a fourth one or more objects of the application, and wherein a first connection is generated to connect the first correlation with the second correlation, and wherein a second connection is generated to connect the second correlation with the fourth correlation, and wherein the first correlation connected with the second correlation and the second correlation connected with the fourth correlation form at least a portion of a knowledge structure or a knowledgebase.

35. (new) A non-transitory machine readable medium having a program stored thereon that when executed by one or more processor circuits causes the one or more processor circuits to perform operations comprising:

accessing a memory that stores at least a first correlation including a first one or more object representations correlated with a first one or more instruction sets for operating a first avatar of an application and a second correlation including a second one or more object representations correlated with a second one or more instruction sets for operating the first avatar of the application, wherein the first one or more object representations represent a first one or more objects of the application and the second one or more object representations represent a second one or more objects of the application, and wherein at least a portion of the first correlation and at least a portion of the second correlation are learned in a learning process while the first avatar of the application is at least partially operated by a user;

generating or receiving a third one or more object representations, wherein the third one or more object representations represent a third one or more objects of the application;

determining the first one or more instruction sets for operating the first avatar of the application based on at least partial match between the third one or more object representations and the first one or more object representations; and

in response to the determining, causing the first avatar of the application or a second avatar of the application to autonomously perform one or more operations defined by the first one or more instruction sets for operating the first avatar of the application at least by causing the one or more processor circuits or another one or more processor circuits to execute the first one or more instruction sets for operating the first avatar of the application.

36. (new) The non-transitory machine readable medium of claim 35, wherein, to generate the first correlation, a determination is made that the first one or more instruction sets for operating the first avatar of the application temporally correspond to the first one or more object representations, and wherein, to generate the second correlation, a determination is made that the second one or more instruction sets for operating the first avatar of the application temporally correspond to the second one or more object representations.

37. (new) The non-transitory machine readable medium of claim 35, wherein the first one or more objects of the application include one or more objects of the application in the first avatar's surrounding, and wherein the third one or more objects of the application include one or more objects of the application in the second avatar's surrounding, and wherein the first one or more instruction sets for operating the first avatar of the application are applied to the second avatar of the application so that the second avatar of the application autonomously performs the one or more operations defined by the first one or more instruction sets for operating the first avatar of the application.

38. (new) A method comprising:

(a) accessing a memory that stores at least a first correlation including a first one or more object representations correlated with a first one or more instruction sets for operating a first avatar of an application and a second

correlation including a second one or more object representations correlated with a second one or more instruction sets for operating the first avatar of the application, wherein the first one or more object representations represent a first one or more objects of the application and the second one or more object representations represent a second one or more objects of the application, and wherein at least a portion of the first correlation and at least a portion of the second correlation are learned in a learning process while the first avatar of the application is at least partially operated by a user, the accessing of (a) performed by one or more processor circuits;

(b) generating or receiving a third one or more object representations, wherein the third one or more object representations represent a third one or more objects of the application, the generating or the receiving of (b) performed by the one or more processor circuits;

(c) determining the first one or more instruction sets for operating the first avatar of the application based on at least partial match between the third one or more object representations and the first one or more object representations, the determining of (c) performed by the one or more processor circuits;

(d) executing the first one or more instruction sets for operating the first avatar of the application, the executing of (d) performed by the one or more processor circuits or by another one or more processor circuits in response to the determining of (c); and

(e) performing, by the first avatar of the application or by a second avatar of the application, one or more operations defined by the first one or more instruction sets for operating the first avatar of the application.

39. (new) The method of claim 38, wherein, to generate the first correlation, a determination is made that the first one or more instruction sets for operating the first avatar of the application temporally correspond to the first one or more object representations, and wherein, to generate the second correlation, a determination is made that the second one or more instruction sets for operating the first avatar of the application temporally correspond to the second one or more object representations.

40. (new) The method of claim 38, wherein the first one or more objects of the application include one or more objects of the application in the first avatar's surrounding, and wherein the third one or more objects of the application include one or more objects of the application in the second avatar's surrounding, and wherein the first one or more instruction sets for operating the first avatar of the application are applied to the second avatar of the application so that the second avatar of the application autonomously performs the one or more operations defined by the first one or more instruction sets for operating the first avatar of the application.

Remarks

In this preliminary amendment, the applicant cancels original claims 1-20 and presents for examination new claims 21-40. After entry of this preliminary amendment, claims 21-40 are pending. The undersigned may be contacted at (317) 772-1312 concerning this application.

I hereby certify that this correspondence is being submitted electronically via EFS Web to the United States Patent and Trademark Office.

By /Jasmin Cosic/
Jasmin Cosic

Date submitted: April 25, 2019

Respectfully submitted,

/Jasmin Cosic/

Jasmin Cosic

Electronic Acknowledgement Receipt

EFS ID:	35835857
Application Number:	15382743
International Application Number:	
Confirmation Number:	4742
Title of Invention:	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING AN AVATAR'S CIRCUMSTANCES FOR AUTONOMOUS AVATAR OPERATION
First Named Inventor/Applicant Name:	Jasmin Cosic
Customer Number:	116094
Filer:	Jasmin Cosic
Filer Authorized By:	
Attorney Docket Number:	
Receipt Date:	25-APR-2019
Filing Date:	19-DEC-2016
Time Stamp:	17:07:54
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	no

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1		PRELIMINARY_AMENDMENT.pdf	48666 7dbf016688fb5e86758c164b27c8d44a27bca2da	yes	13

Multipart Description/PDF files in .zip description		
Document Description	**Start**	**End**
Preliminary Amendment	1	1
Claims	2	12
Applicant Arguments/Remarks Made in an Amendment	13	13

Warnings:

Information:

Total Files Size (in bytes):	48666

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.
National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.
New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS Remove

Examiner Initial*	Cite No	Patent Number	Kind Code[1]	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	8996432		2015-03-31	Fu; Jicheng	
	2	6754631		2004-06-22	Din	
	3	9305216		2016-04-05	Mishra	
	4	5560011		1996-09-23	Uyama	
	5	6842877		2005-01-11	Robarts , et al.	
	6	7565340		2009-07-21	Herlocker , et al.	
	7	8261199		2012-09-04	Cradick , et al.	
	8	8266608		2012-09-11	Hecht , et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	9	9268454		2016-02-23	Hamilton, II , et al.	

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS

Remove

Examiner Initial*	Cite No	Publication Number	Kind Code1	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	20120290347		2012-11-15	ELAZOUNI; ASHRAF ; et al.	
	2	20150324685		2015-11-12	Bohn; Richard Esten ; et al.	
	3	20100114746		2010-05-06	Bobbitt; Russell Patrick ; et al.	
	4	20120284026		2012-11-08	Cardillo; Peter S. ; et al.	
	5	20140211988		2014-07-31	Fan; Quanfu ; et al.	
	6	20150006171		2015-01-01	Westby; Michael C. ; et al.	
	7	20150264306		2015-09-17	Marilly; Emmanuel ; et al.	
	8	20110085734		2011-04-14	Berg; Jared S. ; et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	9	20150269415		2015-09-24	Gelbman; Alexander	
	10	20160274187		2016-09-22	MENON; SANKARAN M. ; et al.	
	11	20150339213		2015-11-22	LEE; Christopher Stephen ; et al.	
	12	20150310041		2015-10-29	Kier; Scott ; et al.	
	13	20160328480		2016-11-10	Owens; Erich James ; et al.	
	14	20140075249		2014-03-13	SATO; Shuhei ; et al.	
	15	20110270794		2011-11-03	Drory; Tal ; et al.	
	16	20070050719		2007-03-01	Lui; Philip ; et al.	
	17	20070061735		2007-03-15	Hoffberg; Steven M. ; et al.	
	18	20060184410		2017-08-17	Ramamurthy; Shankar ; et al.	
	19	20060265406		2006-11-23	Chkodrov; Gueorgui B. ; et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	20	20090287643		2009-11-19	Corville; Allen O. ; et al.	
	21	20110007079		2011-01-13	Perez et al.	
	22	20070050606		2007-03-01	Ferren et al.	
	23	20090067727		2009-03-12	HIROHATA; Hitoshi	
	24	20130007532		2013-01-03	Miller et al.	
	25	20050240412		2005-10-27	Fujita	
	26	20160167226		2016-06-16	Schnittman	
	27	20090136095		2009-05-28	Marcon et al.	
	28	20080215508		2008-09-04	Hanneman; Jeffrey E. ; et al.	
	29	20090044113		2009-02-12	Jones; Scott T. ; et al.	

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

Examiner Initial*	Cite No	Foreign Document Number3	Country Code2 i	Kind Code4	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T5
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T5
	1	Chen et al. "Case-Based Reasoning System and Artificial Neural Networks: A Review Neural Comput & Applic (2001) 10: pp 264-276, 13 pages	
	2	JOHN J. GREFENSTETTE, CONNIE LOGGIA RAMSEY, ALAN C. SCHULTZ, Learning Sequential Decision Rules Using Simulation Models and Competition, 1990, Navy Center for Applied Research in Artificial Intelligence, Naval Research Laboratory, Washington, DC, 27 pages	
	3	ALAN C. SCHULTZ, JOHN J. GREFENSTETTE, Using a Genetic Algorithm to Learn Behaviors for Autonomous Vehicles, 1992, Navy Center for Applied Research in Artificial Intelligence, Naval Research Laboratory, Washington, DC, 12 pages	
	4	Koppula et al., "Anticipating human activities using object affordances for reactive robotic response", IEEE TRAMI 2016, published 5 May 2015, 16 pages	
	5	Orme, "System design tips for entry level smartphones - part 3", found online at "https://community.arm.com/processors/b/blog/posts/system-design-tips-for-entry-level-smartphones---part-3", Oct 21, 2013, 12 pages	
	6	NOWOSTAWSKI, MARIUSZ et al., "Dynamic Demes Parallel Genetic Algorithm", May 13, 1999, 6 pages	
	7	Stack Overflow, How can I quantify difference between two images?, accessed 2 August 2017 at https://stackoverflow.com/questions/189943/how-can-i-quantify-difference-between-two-images, 8 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	8	SKOLICKI, ZBIGNIEW et al., "The Influence of Migration Sizes and Intervals on Island Models", June 29, 2005, 8 pages	
	9	Vahid Lari, et al., "Decentralized dynamic resource management support for massively parallel processor arrays", September 11, 2011, 8 pages	
	10	Vahid Lari, et al., "Distributed resource reservation in massively parallel processor arrays", May 16, 2011, 4 pages	
	11	Mohsen Hayati and Yazdan Shirvany, "Artificial Neural Network Approach for Short Term Load Forecasting for Illam Region", January 2007, 5 pages	
	12	Hasim Sak, Andrew Senior, and Francoise Beaufays, "Long Short-Term Memory Recurrent Neural Network Architectures for Large Scale Acoustic Modeling", January 2014, 5 pages	
	13	Jorg Walter and Klaus Schulten, "Implementation of self-organizing neural networks for visuo-motor control of an industrial robot", January 1993, 10 pages	
	14	Heikki Hyotyniemi and Aarno Lehtola, "A Universal Relation Database Interface for Knowledge Based Systems", April 1991, 5 pages	
	15	Mrissa, Michael, et al. "An avatar architecture for the web of things." IEEE Internet Computing 19.2 (2015): 30-38., 9 pages	
	16	Luck, Michael, and Ruth Aylett. "Applying artificial intelligence to virtual reality: Intelligent virtual environments." Applied Artificial Intelligence 14.1 (2000): 3-32., 30 pages	
	17	Terdjimi, Mehdi, et al. "An avatar-based adaptation workflow for the web of things." Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), 2016 IEEE 25th International Conference on. IEEE, 2016., 6 pages	
	18	Bogdanovych, Anton, et al. "Authentic interactive reenactment of cultural heritage with 3D virtual worlds and artificial intelligence." Applied Artificial Intelligence 24.6 (2010): 617-647., 32 pages	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

	19	Hernandez, Marco E. Perez, and Stephan Reiff-Marganiec. "Autonomous and self controlling smart objects for the future internet." Future internet of things and cloud (FiCloud), 2015 3rd international conference on. IEEE, 2015., 8 pages	
	20	Medini et al., "Building a Web of Things with Avatars", Managing the Web of Things (2017), 30 pages	

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature		Date Considered	

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	
	Examiner Name	
	Attorney Docket Number	

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Jasmin Cosic/	Date (YYYY-MM-DD)	2019-07-02
Name/Print	Jasmin Cosic	Registration Number	

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Electronic Acknowledgement Receipt

EFS ID:	36475505
Application Number:	15382743
International Application Number:	
Confirmation Number:	4742
Title of Invention:	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING AN AVATAR'S CIRCUMSTANCES FOR AUTONOMOUS AVATAR OPERATION
First Named Inventor/Applicant Name:	Jasmin Cosic
Customer Number:	116094
Filer:	Jasmin Cosic
Filer Authorized By:	
Attorney Docket Number:	
Receipt Date:	02-JUL-2019
Filing Date:	19-DEC-2016
Time Stamp:	12:17:25
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	no

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1	Information Disclosure Statement (IDS) Form (SB08)	IDS_Part6.pdf	618679 93ecc43eeeab415174998b4a41b82c14796af5a8	no	9

Warnings:

Information:

2	Non Patent Literature	NPL_Part6.pdf	25533925 58bed29843cae9904ec69a4efe47cc83b3ece30a	no	254

Warnings:

The page size in the PDF is too large. The pages should be 8.5 x 11 or A4. If this PDF is submitted, the pages will be resized upon entry into the Image File Wrapper and may affect subsequent processing

Information:

Total Files Size (in bytes):	26152604

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.
National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.
New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

PTO/SB/06 (09-11)
Approved for use through 1/31/2014. OMB 0651-0032
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

PATENT APPLICATION FEE DETERMINATION RECORD Substitute for Form PTO-875	Application or Docket Number 15/382,743	Filing Date 12/19/2016	☐ To be Mailed

ENTITY: ☐ LARGE ☑ SMALL ☐ MICRO

APPLICATION AS FILED - PART I

FOR	(Column 1) NUMBER FILED	(Column 2) NUMBER EXTRA	RATE ($)	FEE ($)
☐ BASIC FEE (37 CFR 1.16(a), (b), or (c))	N/A	N/A	N/A	
☐ SEARCH FEE (37 CFR 1.16(k), (i), or (m))	N/A	N/A	N/A	
☐ EXAMINATION FEE (37 CFR 1.16(o), (p), or (q))	N/A	N/A	N/A	
TOTAL CLAIMS (37 CFR 1.16(i))	minus 20 =	*	x $40 =	
INDEPENDENT CLAIMS (37 CFR 1.16(h))	minus 3 =	*	x $210 =	
☐ APPLICATION SIZE FEE (37 CFR 1.16(s))	If the specification and drawings exceed 100 sheets of paper, the application size fee due is $310 ($155 for small entity) for each additional 50 sheets or fraction thereof. See 35 U.S.C. 41(a)(1)(G) and 37 CFR 1.16(s).			
☐ MULTIPLE DEPENDENT CLAIM PRESENT (37 CFR 1.16(j))				
* If the difference in column 1 is less than zero, enter "0" in column 2.			TOTAL	

APPLICATION AS AMENDED - PART II

AMENDMENT	08/10/2019	(Column 1) CLAIMS REMAINING AFTER AMENDMENT		(Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR	(Column 3) PRESENT EXTRA	RATE ($)	ADDITIONAL FEE ($)
	Total (37 CFR 1.16(i))	* 20	Minus	** 20	= 0	x $50 =	0
	Independent (37 CFR 1.16(h))	* 3	Minus	*** 3	= 0	x $230 =	0
	☐ Application Size Fee (37 CFR 1.16(s))						
	☐ FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))						
						TOTAL ADD'L FEE	0

AMENDMENT		(Column 1) CLAIMS REMAINING AFTER AMENDMENT		(Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR	(Column 3) PRESENT EXTRA	RATE ($)	ADDITIONAL FEE ($)
	Total (37 CFR 1.16(i))	*	Minus	**	=	x $0 =	
	Independent (37 CFR 1.16(h))	*	Minus	***	=	x $0 =	
	☐ Application Size Fee (37 CFR 1.16(s))						
	☐ FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))						
						TOTAL ADD'L FEE	

* If the entry in column 1 is less than the entry in column 2, write "0" in column 3.	SLIE
** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, enter "20".	/SHARON M WEST/
*** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, enter "3".	
The "Highest Number Previously Paid For" (Total or Independent) is the highest number found in the appropriate box in column 1.	

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Inventor:	Jasmin Cosic	Confirmation No.: 4742
Title:	"ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING AN AVATAR'S CIRCUMSTANCES FOR AUTONOMOUS AVATAR OPERATION"	
Serial No.:	15/382,743	Filed: December 19, 2016
Examiner:	SCHNEE, HAL W	Group Art Unit: 2125

Via EFS-Web August 10, 2019

Mail Stop Amendment
COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, VA 22313-1450

PRELIMINARY AMENDMENT

Dear Commissioner:

Before examination on the merits, please amend the claims in the above-identified application as shown in the following listing of claims. This preliminary amendment supersedes the preliminary amendment filed on April 25, 2019, therefore, the claims in the following listing of claims should be used for the examination on the merits instead of the claims in the preliminary amendment filed on April 25, 2019.

Inventor: Jasmin Cosic
Serial No.: 15/382,743
Filing Date: December 19, 2016

Listing of Claims

1 - 40 (canceled)

41. (new) A system comprising:

one or more processors; and

one or more non-transitory machine readable media storing machine readable code that, when executed by the one or more processors, causes the one or more processors to perform at least:

accessing a first knowledge cell including a first one or more object representations correlated with a first one or more instruction sets for operating a first avatar of an application, wherein the first one or more object representations represent one or more objects of the application;

generating or receiving a second one or more object representations, wherein the second one or more object representations represent one or more objects of the application;

anticipating the first one or more instruction sets for operating the first avatar of the application included in the first knowledge cell based on at least partial match between the second one or more object representations and the first one or more object representations included in the first knowledge cell; and

at least in response to the anticipating, causing the first avatar of the application or a second avatar of the application to perform one or more

operations defined by the first one or more instruction sets for operating the first avatar of the application included in the first knowledge cell at least by executing the first one or more instruction sets for operating the first avatar of the application included in the first knowledge cell.

42. (new) The system of claim 41, wherein the first one or more object representations include: a stream of one or more object representations, a collection of object representations, or a stream of collections of object representations, and wherein the second one or more object representations include: a stream of one or more object representations, a collection of object representations, or a stream of collections of object representations.

43. (new) The system of claim 41, wherein the first one or more instruction sets for operating the first avatar of the application included in the first knowledge cell include one or more instruction sets for operating a portion of the first avatar of the application, and wherein the causing the first avatar of the application or the second avatar of the application to perform the one or more operations defined by the first one or more instruction sets for operating the first avatar of the application included in the first knowledge cell includes causing the portion of the first avatar of the application or a portion of the second avatar of the application to perform one or more operations defined by the one or more instruction sets for operating the portion of the first avatar of the application.

44. (new) The system of claim 41, wherein the executing the first one or more instruction sets for operating the first avatar of the application included in the first knowledge cell is performed via an execution interface.

45. (new) The system of claim 41, wherein the first one or more instruction sets for operating the first avatar of the application included in the first knowledge cell are obtained via an acquisition interface.

46. (new) The system of claim 41, wherein the one or more objects of the application represented by the first one or more object representations include one or more objects of the application in the first avatar's surrounding, and wherein the first avatar's surrounding includes at least one selected from the group comprising:

a part of the application in an area of interest around the first avatar,

a part of the application defined by a threshold distance from the first avatar,

a part of the application relative to the first avatar,

a part of the application around the first avatar,

a part of the application that is shown to the user, and

a part of the application.

47. (new) The system of claim 41, wherein at least one object of the application of the one or more objects of the application represented by the first one or more

object representations is the same as at least one object of the application of the one or more objects of the application represented by the second one or more object representations, or at least one object of the application of the one or more objects of the application represented by the first one or more object representations is different than at least one object of the application of the one or more objects of the application represented by the second one or more object representations.

48. (new) The system of claim 41, wherein at least some elements of the system are included in: a single device, or multiple devices, and wherein the one or more processors include: one or more microcontrollers, one or more computing devices, or one or more electronic devices, and wherein the first knowledge cell is stored in or on one selected from the group comprising: at least one non-transitory machine readable medium of the one or more non-transitory machine readable media, another one or more non-transitory machine readable media, one or more volatile memories, one or more non-volatile memories, one or more storage devices, and one or more storage systems, and wherein the application includes one or more versions of the application, one or more upgrades of the application, one or more sequels of the application, one or more instances of the application, or one or more variations of the application, and wherein the first avatar of the application is a first object of the application and the second avatar of the application is a second object of the application.

49. (new) The system of claim 41, wherein the first knowledge cell is a data structure for storing, structuring, or organizing the first one or more object representations correlated with the first one or more instruction sets for operating the first avatar of the application.

50. (new) A method implemented using a computing system that includes one or more processors, the method comprising:

accessing a memory that stores at least a first knowledge cell including a first one or more object representations correlated with a first one or more instruction sets for operating a first avatar of an application, wherein the first one or more object representations represent one or more objects of the application;

generating or receiving a second one or more object representations, wherein the second one or more object representations represent one or more objects of the application;

anticipating the first one or more instruction sets for operating the first avatar of the application included in the first knowledge cell based on at least partial match between the second one or more object representations and the first one or more object representations included in the first knowledge cell; and

at least in response to the anticipating, executing the first one or more instruction sets for operating the first avatar of the application included in the first knowledge cell, wherein the first avatar of the application or a second avatar of the application performs one or more operations defined by the first one or more

instruction sets for operating the first avatar of the application included in the first knowledge cell.

51. (new) The method of claim 50, wherein the first one or more object representations include: a stream of one or more object representations, a collection of object representations, or a stream of collections of object representations, and wherein the second one or more object representations include: a stream of one or more object representations, a collection of object representations, or a stream of collections of object representations.

52. (new) The method of claim 50, wherein the first one or more instruction sets for operating the first avatar of the application included in the first knowledge cell include one or more instruction sets for operating a portion of the first avatar of the application, and wherein the performing, by the first avatar of the application or by the second avatar of the application, the one or more operations defined by the first one or more instruction sets for operating the first avatar of the application included in the first knowledge cell includes performing, by the portion of the first avatar of the application or by a portion of the second avatar of the application, one or more operations defined by the one or more instruction sets for operating the portion of the first avatar of the application.

53. (new) The method of claim 50, wherein the executing the first one or more instruction sets for operating the first avatar of the application included in the first knowledge cell is performed via an execution interface.

54. (new) The method of claim 50, wherein the first one or more instruction sets for operating the first avatar of the application included in the first knowledge cell are obtained via an acquisition interface.

55. (new) The method of claim 50, wherein the one or more objects of the application represented by the first one or more object representations include one or more objects of the application in the first avatar's surrounding, and wherein the first avatar's surrounding includes at least one selected from the group comprising:

a part of the application in an area of interest around the first avatar,

a part of the application defined by a threshold distance from the first avatar,

a part of the application relative to the first avatar,

a part of the application around the first avatar,

a part of the application that is shown to the user, and

a part of the application.

56. (new) The method of claim 50, wherein at least one object of the application of the one or more objects of the application represented by the first one or more

object representations is the same as at least one object of the application of the one or more objects of the application represented by the second one or more object representations, or at least one object of the application of the one or more objects of the application represented by the first one or more object representations is different than at least one object of the application of the one or more objects of the application represented by the second one or more object representations.

57. (new) The method of claim 50, wherein at least some elements of the computing system are included in: a single device, or multiple devices, and wherein the one or more processors include: one or more microcontrollers, one or more computing devices, or one or more electronic devices, and wherein the memory includes: a volatile memory, or a non-volatile memory, and wherein the application includes one or more versions of the application, one or more upgrades of the application, one or more sequels of the application, one or more instances of the application, or one or more variations of the application, and wherein the first avatar of the application is a first object of the application and the second avatar of the application is a second object of the application.

58. (new) The method of claim 50, wherein the first knowledge cell is a data structure for storing, structuring, or organizing the first one or more object representations correlated with the first one or more instruction sets for operating the first avatar of the application.

59. (new) A system comprising:

a memory that stores at least a first knowledge cell including a first one or more object representations correlated with a first one or more instruction sets for operating a first avatar of an application, wherein the first one or more object representations represent one or more objects of the application;

means for generating or receiving a second one or more object representations, wherein the second one or more object representations represent one or more objects of the application;

means for anticipating the first one or more instruction sets for operating the first avatar of the application included in the first knowledge cell based on at least partial match between the second one or more object representations and the first one or more object representations included in the first knowledge cell; and

means for executing, at least in response to the anticipating, the first one or more instruction sets for operating the first avatar of the application included in the first knowledge cell, wherein the first avatar of the application or a second avatar of the application performs one or more operations defined by the first one or more instruction sets for operating the first avatar of the application included in the first knowledge cell.

60. (new) The system of claim 59, wherein the means for generating or receiving the second one or more object representations includes one or more

processors, and wherein the means for anticipating the first one or more instruction sets for operating the first avatar of the application included in the first knowledge cell based on the at least partial match between the second one or more object representations and the first one or more object representations included in the first knowledge cell includes one or more processors, and wherein the means for executing, at least in response to the anticipating, the first one or more instruction sets for operating the first avatar of the application included in the first knowledge cell includes one or more processors.

Inventor: Jasmin Cosic
Serial No.: 15/382,743
Filing Date: December 19, 2016

Remarks

In this preliminary amendment, the applicant cancels claims 1-40, and presents for examination new claims 41-60. After entry of this preliminary amendment, claims 41-60 are pending. The undersigned may be contacted at (317) 772-1312 concerning this application.

I hereby certify that this correspondence is being submitted electronically via EFS Web to the United States Patent and Trademark Office.

By /Jasmin Cosic/
Jasmin Cosic

Date submitted: August 10, 2019

Respectfully submitted,

/Jasmin Cosic/

Jasmin Cosic

Electronic Acknowledgement Receipt

EFS ID:	36840291
Application Number:	15382743
International Application Number:	
Confirmation Number:	4742
Title of Invention:	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING AN AVATAR'S CIRCUMSTANCES FOR AUTONOMOUS AVATAR OPERATION
First Named Inventor/Applicant Name:	Jasmin Cosic
Customer Number:	116094
Filer:	Jasmin Cosic
Filer Authorized By:	
Attorney Docket Number:	
Receipt Date:	10-AUG-2019
Filing Date:	19-DEC-2016
Time Stamp:	22:51:07
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	no

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1		PRELIMINARY_AMENDMENT.pdf	44639 05f3eca3b7499f0072fa05ade1d9cb7c5d2dc8ee	yes	12

Multipart Description/PDF files in .zip description		
Document Description	**Start**	**End**
Preliminary Amendment	1	1
Claims	2	11
Applicant Arguments/Remarks Made in an Amendment	12	12

Warnings:

Information:

Total Files Size (in bytes):	44639

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.

National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

Doc Code: DIST.E.FILE **Document Description: Electronic Terminal Disclaimer - Filed**	PTO/SB/26 U.S. Patent and Trademark Office Department of Commerce

Electronic Petition Request	**TERMINAL DISCLAIMER TO OBVIATE A DOUBLE PATENTING REJECTION OVER A "PRIOR" PATENT**
Application Number	15382743
Filing Date	19-Dec-2016
First Named Inventor	Jasmin Cosic
Attorney Docket Number	
Title of Invention	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING AN AVATAR'S CIRCUMSTANCES FOR AUTONOMOUS AVATAR OPERATION

☒ Filing of terminal disclaimer does not obviate requirement for response under 37 CFR 1.111 to outstanding Office Action

☒ This electronic Terminal Disclaimer is not being used for a Joint Research Agreement.

Owner	Percent Interest
Jasmin Cosic	100%

The owner(s) with percent interest listed above in the instant application hereby disclaims, except as provided below, the terminal part of the statutory term of any patent granted on the instant application which would extend beyond the expiration date of the full statutory term of prior patent number(s)

10402731

as the term of said prior patent is presently shortened by any terminal disclaimer. The owner hereby agrees that any patent so granted on the instant application shall be enforceable only for and during such period that it and the prior patent are commonly owned. This agreement runs with any patent granted on the instant application and is binding upon the grantee, its successors or assigns.

In making the above disclaimer, the owner does not disclaim the terminal part of the term of any patent granted on the instant application that would extend to the expiration date of the full statutory term of the prior patent, "as the term of said prior patent is presently shortened by any terminal disclaimer," in the event that said prior patent later:
- expires for failure to pay a maintenance fee;
- is held unenforceable;
- is found invalid by a court of competent jurisdiction;
- is statutorily disclaimed in whole or terminally disclaimed under 37 CFR 1.321;
- has all claims canceled by a reexamination certificate;
- is reissued; or
- is in any manner terminated prior to the expiration of its full statutory term as presently shortened by any terminal disclaimer.

◉ Terminal disclaimer fee under 37 CFR 1.20(d) is included with Electronic Terminal Disclaimer request.

○ I certify, in accordance with 37 CFR 1.4(d)(4), that the terminal disclaimer fee under 37 CFR 1.20(d) required for this terminal disclaimer has already been paid in the above-identified application.

Applicant claims the following fee status:

◉ Small Entity

○ Micro Entity

○ Regular Undiscounted

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

THIS PORTION MUST BE COMPLETED BY THE SIGNATORY OR SIGNATORIES

I certify, in accordance with 37 CFR 1.4(d)(4) that I am:

○ An attorney or agent registered to practice before the Patent and Trademark Office who is of record in this application

Registration Number ________

◉ A sole inventor

○ A joint inventor; I certify that I am authorized to sign this submission on behalf of all of the inventors as evidenced by the power of attorney in the application

○ A joint inventor; all of whom are signing this request

Signature	/Jasmin Cosic/
Name	Jasmin Cosic

*Statement under 37 CFR 3.73(b) is required if terminal disclaimer is signed by the assignee (owner).
Form PTO/SB/96 may be used for making this certification. See MPEP § 324.

Electronic Patent Application Fee Transmittal

Application Number:	15382743
Filing Date:	19-Dec-2016
Title of Invention:	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING AN AVATAR'S CIRCUMSTANCES FOR AUTONOMOUS AVATAR OPERATION
First Named Inventor/Applicant Name:	Jasmin Cosic
Filer:	Jasmin Cosic
Attorney Docket Number:	

Filed as Small Entity

Filing Fees for **Utility under 35 USC 111(a)**

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Basic Filing:				
STATUTORY OR TERMINAL DISCLAIMER	2814	1	160	160
Pages:				
Claims:				
Miscellaneous-Filing:				
Petition:				
Patent-Appeals-and-Interference:				
Post-Allowance-and-Post-Issuance:				

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Extension-of-Time:				
Miscellaneous:				
	Total in USD ($)			**160**

Application Number * 15/382,743 *	Application/Control No. 15/382,743	Applicant(s)/Patent under Reexamination Cosic, Jasmin	
	Examiner SCHNEE, HAL W	Art Unit 2125	

Document Code - DISQ	**Internal Document - DO NOT MAIL**

TERMINAL DISCLAIMER	☑ APPROVED	☐ DISAPPROVED
Date Filed: **24 November 2019**	**This patent is subject to a Terminal Disclaimer**	

Approved/Disapproved by:
/JEAN PROCTOR/ **Technology Center: OPLC** **Telephone: (571)272-1040**

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

NOTICE OF ALLOWANCE AND FEE(S) DUE

116094 7590 12/11/2019

Jasmin Cosic
108 Woodbury Street
Pawtucket, RI 02861

EXAMINER	
SCHNEE, HAL W	
ART UNIT	PAPER NUMBER
2125	

DATE MAILED: 12/11/2019

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
15/382,743	12/19/2016	Jasmin Cosic		4742

TITLE OF INVENTION: ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING AN AVATAR'S CIRCUMSTANCES FOR AUTONOMOUS AVATAR OPERATION

APPLN. TYPE	ENTITY STATUS	ISSUE FEE DUE	PUBLICATION FEE DUE	PREV. PAID ISSUE FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	SMALL	$500	$0.00	$0.00	$500	03/11/2020

THE APPLICATION IDENTIFIED ABOVE HAS BEEN EXAMINED AND IS ALLOWED FOR ISSUANCE AS A PATENT. PROSECUTION ON THE MERITS IS CLOSED. THIS NOTICE OF ALLOWANCE IS NOT A GRANT OF PATENT RIGHTS. THIS APPLICATION IS SUBJECT TO WITHDRAWAL FROM ISSUE AT THE INITIATIVE OF THE OFFICE OR UPON PETITION BY THE APPLICANT. SEE 37 CFR 1.313 AND MPEP 1308.

THE ISSUE FEE AND PUBLICATION FEE (IF REQUIRED) MUST BE PAID WITHIN THREE MONTHS FROM THE MAILING DATE OF THIS NOTICE OR THIS APPLICATION SHALL BE REGARDED AS ABANDONED. THIS STATUTORY PERIOD CANNOT BE EXTENDED. SEE 35 U.S.C. 151. THE ISSUE FEE DUE INDICATED ABOVE DOES NOT REFLECT A CREDIT FOR ANY PREVIOUSLY PAID ISSUE FEE IN THIS APPLICATION. IF AN ISSUE FEE HAS PREVIOUSLY BEEN PAID IN THIS APPLICATION (AS SHOWN ABOVE), THE RETURN OF PART B OF THIS FORM WILL BE CONSIDERED A REQUEST TO REAPPLY THE PREVIOUSLY PAID ISSUE FEE TOWARD THE ISSUE FEE NOW DUE.

HOW TO REPLY TO THIS NOTICE:

I. Review the ENTITY STATUS shown above. If the ENTITY STATUS is shown as SMALL or MICRO, verify whether entitlement to that entity status still applies.

If the ENTITY STATUS is the same as shown above, pay the TOTAL FEE(S) DUE shown above.

If the ENTITY STATUS is changed from that shown above, on PART B - FEE(S) TRANSMITTAL, complete section number 5 titled "Change in Entity Status (from status indicated above)".

For purposes of this notice, small entity fees are 1/2 the amount of undiscounted fees, and micro entity fees are 1/2 the amount of small entity fees.

II. PART B - FEE(S) TRANSMITTAL, or its equivalent, must be completed and returned to the United States Patent and Trademark Office (USPTO) with your ISSUE FEE and PUBLICATION FEE (if required). If you are charging the fee(s) to your deposit account, section "4b" of Part B - Fee(s) Transmittal should be completed and an extra copy of the form should be submitted. If an equivalent of Part B is filed, a request to reapply a previously paid issue fee must be clearly made, and delays in processing may occur due to the difficulty in recognizing the paper as an equivalent of Part B.

III. All communications regarding this application must give the application number. Please direct all communications prior to issuance to Mail Stop ISSUE FEE unless advised to the contrary.

IMPORTANT REMINDER: Maintenance fees are due in utility patents issuing on applications filed on or after Dec. 12, 1980. It is patentee's responsibility to ensure timely payment of maintenance fees when due. More information is available at www.uspto.gov/PatentMaintenanceFees.

PART B - FEE(S) TRANSMITTAL

Complete and send this form, together with applicable fee(s), by mail or fax, or via EFS-Web.

By mail, send to: Mail Stop ISSUE FEE
Commissioner for Patents
P.O. Box 1450
Alexandria, Virginia 22313-1450

By fax, send to: (571)-273-2885

INSTRUCTIONS: This form should be used for transmitting the ISSUE FEE and PUBLICATION FEE (if required). Blocks 1 through 5 should be completed where appropriate. All further correspondence including the Patent, advance orders and notification of maintenance fees will be mailed to the current correspondence address as indicated unless corrected below or directed otherwise in Block 1, by (a) specifying a new correspondence address; and/or (b) indicating a separate "FEE ADDRESS" for maintenance fee notifications.

CURRENT CORRESPONDENCE ADDRESS (Note: Use Block 1 for any change of address)

116094 7590 12/11/2019

Jasmin Cosic
108 Woodbury Street
Pawtucket, RI 02861

Note: A certificate of mailing can only be used for domestic mailings of the Fee(s) Transmittal. This certificate cannot be used for any other accompanying papers. Each additional paper, such as an assignment or formal drawing, must have its own certificate of mailing or transmission.

Certificate of Mailing or Transmission

I hereby certify that this Fee(s) Transmittal is being deposited with the United States Postal Service with sufficient postage for first class mail in an envelope addressed to the Mail Stop ISSUE FEE address above, or being transmitted to the USPTO via EFS-Web or by facsimile to (571) 273-2885, on the date below.

(Typed or printed name)

(Signature)

(Date)

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
15/382,743	12/19/2016	Jasmin Cosic		4742

TITLE OF INVENTION: ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING AN AVATAR'S CIRCUMSTANCES FOR AUTONOMOUS AVATAR OPERATION

APPLN. TYPE	ENTITY STATUS	ISSUE FEE DUE	PUBLICATION FEE DUE	PREV. PAID ISSUE FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	SMALL	$500	$0.00	$0.00	$500	03/11/2020

EXAMINER	ART UNIT	CLASS-SUBCLASS
SCHNEE, HAL W	2125	706-011000

1. Change of correspondence address or indication of "Fee Address" (37 CFR 1.363).

❏ Change of correspondence address (or Change of Correspondence Address form PTO/SB/122) attached.

❏ "Fee Address" indication (or "Fee Address" Indication form PTO/SB/47; Rev 03-09 or more recent) attached. **Use of a Customer Number is required.**

2. For printing on the patent front page, list
(1) The names of up to 3 registered patent attorneys or agents OR, alternatively,
(2) The name of a single firm (having as a member a registered attorney or agent) and the names of up to 2 registered patent attorneys or agents. If no name is listed, no name will be printed.

1____________________
2____________________
3____________________

3. ASSIGNEE NAME AND RESIDENCE DATA TO BE PRINTED ON THE PATENT (print or type)

PLEASE NOTE: Unless an assignee is identified below, no assignee data will appear on the patent. If an assignee is identified below, the document must have been previously recorded, or filed for recordation, as set forth in 37 CFR 3.11 and 37 CFR 3.81(a). Completion of this form is NOT a substitute for filing an assignment.

(A) NAME OF ASSIGNEE

(B) RESIDENCE: (CITY and STATE OR COUNTRY)

Please check the appropriate assignee category or categories (will not be printed on the patent) : ❏ Individual ❏ Corporation or other private group entity ❏ Government

4a. Fees submitted: ❏Issue Fee ❏Publication Fee (if required) ❏Advance Order - # of Copies __________

4b. Method of Payment: *(Please first reapply any previously paid fee shown above)*

❏ Electronic Payment via EFS-Web ❏ Enclosed check ❏ Non-electronic payment by credit card (Attach form PTO-2038)

❏ The Director is hereby authorized to charge the required fee(s), any deficiency, or credit any overpayment to Deposit Account No. __________

5. **Change in Entity Status** (from status indicated above)

❏ Applicant certifying micro entity status. See 37 CFR 1.29

❏ Applicant asserting small entity status. See 37 CFR 1.27

❏ Applicant changing to regular undiscounted fee status.

NOTE: Absent a valid certification of Micro Entity Status (see forms PTO/SB/15A and 15B), issue fee payment in the micro entity amount will not be accepted at the risk of application abandonment.
NOTE: If the application was previously under micro entity status, checking this box will be taken to be a notification of loss of entitlement to micro entity status.
NOTE: Checking this box will be taken to be a notification of loss of entitlement to small or micro entity status, as applicable.

NOTE: This form must be signed in accordance with 37 CFR 1.31 and 1.33. See 37 CFR 1.4 for signature requirements and certifications.

Authorized Signature ____________________ Date ____________________

Typed or printed name ____________________ Registration No. ____________________

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
15/382,743	12/19/2016	Jasmin Cosic		4742

116094 7590 12/11/2019

Jasmin Cosic
108 Woodbury Street
Pawtucket, RI 02861

EXAMINER	
SCHNEE, HAL W	
ART UNIT	PAPER NUMBER
2125	

DATE MAILED: 12/11/2019

Determination of Patent Term Adjustment under 35 U.S.C. 154 (b)

(Applications filed on or after May 29, 2000)

The Office has discontinued providing a Patent Term Adjustment (PTA) calculation with the Notice of Allowance.

Section 1(h)(2) of the AIA Technical Corrections Act amended 35 U.S.C. 154(b)(3)(B)(i) to eliminate the requirement that the Office provide a patent term adjustment determination with the notice of allowance. See Revisions to Patent Term Adjustment, 78 Fed. Reg. 19416, 19417 (Apr. 1, 2013). Therefore, the Office is no longer providing an initial patent term adjustment determination with the notice of allowance. The Office will continue to provide a patent term adjustment determination with the Issue Notification Letter that is mailed to applicant approximately three weeks prior to the issue date of the patent, and will include the patent term adjustment on the patent. Any request for reconsideration of the patent term adjustment determination (or reinstatement of patent term adjustment) should follow the process outlined in 37 CFR 1.705.

Any questions regarding the Patent Term Extension or Adjustment determination should be directed to the Office of Patent Legal Administration at (571)-272-7702. Questions relating to issue and publication fee payments should be directed to the Customer Service Center of the Office of Patent Publication at 1-(888)-786-0101 or (571)-272-4200.

OMB Clearance and PRA Burden Statement for PTOL-85 Part B

The Paperwork Reduction Act (PRA) of 1995 requires Federal agencies to obtain Office of Management and Budget approval before requesting most types of information from the public. When OMB approves an agency request to collect information from the public, OMB (i) provides a valid OMB Control Number and expiration date for the agency to display on the instrument that will be used to collect the information and (ii) requires the agency to inform the public about the OMB Control Number's legal significance in accordance with 5 CFR 1320.5(b).

The information collected by PTOL-85 Part B is required by 37 CFR 1.311. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 30 minutes to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, Virginia 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, Virginia 22313-1450. Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether disclosure of these records is required by the Freedom of Information Act.
2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.
3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.
4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).
5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.
6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).
7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.
8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspection or an issued patent.
9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Notice of Allowability	Application No. 15/382,743	Applicant(s) Cosic, Jasmin	
	Examiner HAL W SCHNEE	Art Unit 2125	AIA (FITF) Status Yes

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address--

All claims being allowable, PROSECUTION ON THE MERITS IS (OR REMAINS) CLOSED in this application. If not included herewith (or previously mailed), a Notice of Allowance (PTOL-85) or other appropriate communication will be mailed in due course. **THIS NOTICE OF ALLOWABILITY IS NOT A GRANT OF PATENT RIGHTS.** This application is subject to withdrawal from issue at the initiative of the Office or upon petition by the applicant. See 37 CFR 1.313 and MPEP 1308.

1.☑ This communication is responsive to preliminary amendment filed 10 August 2019.
☐ A declaration(s)/affidavit(s) under **37 CFR 1.130(b)** was/were filed on ______.

2.☐ An election was made by the applicant in response to a restriction requirement set forth during the interview on ______; the restriction requirement and election have been incorporated into this action.

3.☑ The allowed claim(s) is/are 41-60 . As a result of the allowed claim(s), you may be eligible to benefit from the **Patent Prosecution Highway** program at a participating intellectual property office for the corresponding application. For more information, please see **http://www.uspto.gov/patents/init_events/pph/index.jsp** or send an inquiry to **PPHfeedback@uspto.gov.**

4.☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

Certified copies:

a) ☐All b) ☐ Some *c) ☐ None of the:

1. ☐ Certified copies of the priority documents have been received.
2. ☐ Certified copies of the priority documents have been received in Application No. ______ .
3. ☐ Copies of the certified copies of the priority documents have been received in this national stage application from the International Bureau (PCT Rule 17.2(a)).

* Certified copies not received: ______ .

Applicant has THREE MONTHS FROM THE "MAILING DATE" of this communication to file a reply complying with the requirements noted below. Failure to timely comply will result in ABANDONMENT of this application.
THIS THREE-MONTH PERIOD IS NOT EXTENDABLE.

5.☐ CORRECTED DRAWINGS (as "replacement sheets") must be submitted.
☐ including changes required by the attached Examiner's Amendment / Comment or in the Office action of Paper No./Mail Date ______ .

Identifying indicia such as the application number (see 37 CFR 1.84(c)) should be written on the drawings in the front (not the back) of each sheet. Replacement sheet(s) should be labeled as such in the header according to 37 CFR 1.121(d).

6.☐ DEPOSIT OF and/or INFORMATION about the deposit of BIOLOGICAL MATERIAL must be submitted. Note the attached Examiner's comment regarding REQUIREMENT FOR THE DEPOSIT OF BIOLOGICAL MATERIAL.

Attachment(s)

1.☑ Notice of References Cited (PTO-892)
2.☑ Information Disclosure Statements (PTO/SB/08), Paper No./Mail Date 12/20/16 (five documents), 7/2/19.
3.☐ Examiner's Comment Regarding Requirement for Deposit of Biological Material ______.
4.☑ Interview Summary (PTO-413), Paper No./Mail Date. 11/25/19.
5. ☑ Examiner's Amendment/Comment
6. ☑ Examiner's Statement of Reasons for Allowance
7. ☐ Other ______.

/HAL SCHNEE/
Primary Examiner, Art Unit 2125

Notice of Pre-AIA or AIA Status

1. The present application, filed on or after March 16, 2013, is being examined under the first inventor to file provisions of the AIA.

EXAMINER'S AMENDMENT

2. An examiner’s amendment to the record appears below. Should the changes and/or additions be unacceptable to applicant, an amendment may be filed as provided by 37 CFR 1.312. To ensure consideration of such an amendment, it MUST be submitted no later than the payment of the issue fee.

Authorization for this examiner’s amendment was given in an interview with Jasmin Cosic on 25 November 2019.

The claims have been amended as follows:

41. A system comprising:

one or more processors; and

one or more non-transitory machine readable media storing machine readable code that, when executed by the one or more processors, causes the one or more processors to perform at least:

accessing a first ~~knowledge cell~~ correlation including a first one or more object representations correlated with a first one or more instruction sets for operating a first avatar of an application, wherein the first one or more object representations represent one or more objects of the application;

generating or receiving a second one or more object representations, wherein the second one or more object representations represent one or more objects of the application;

~~anticipating~~ determining the first one or more instruction sets for operating the first avatar of the application ~~included in the first knowledge cell~~ based on at least partial match between the second one or more object representations and the first one or more object representations ~~included in the first knowledge cell~~; and

at least in response to the ~~anticipating~~ determining, causing the first avatar of the application or a second avatar of the application to perform one or more operations defined by the first one or more instruction sets for operating the first avatar of the application ~~included in the first knowledge cell~~ at least by executing the first one or more instruction sets for operating the first avatar of the application ~~included in the first knowledge cell~~.

42. The system of claim 41, wherein the first one or more object representations include: one object representation, a stream of ~~one or more~~ object representations, a collection of object representations, or a stream of collections of object representations, and wherein the second one or more object representations include: one object representation, a stream of ~~one or more~~ object representations, a collection of object representations, or a stream of collections of object representations, and wherein the one or more objects of the application represented by the first one or more object representations are detected at a first time or during a first time period, and wherein the one or more objects of the application represented by the second one or more object representations are detected at a second time or during a second time period.

43. The system of claim 41, ~~wherein the first one or more instruction sets for operating the first avatar of the application included in the first knowledge cell include one or more instruction sets for operating a portion of the first avatar of the application, and wherein the causing the first avatar of the application or the second avatar of the application to perform the one or more operations defined by the first one or more instruction sets for operating the first avatar of the application included in the first knowledge cell includes causing the portion of the first avatar of the application or a portion of the second avatar of the application to perform one or more operations defined by the one or more instruction sets for operating the portion of the first avatar of the application.~~ wherein the determining the first one or more instruction sets for operating the first avatar of the application based on the at least partial match between the second one or more object representations and the first one or more object representations includes:

determining that a number of at least partially matching portions of the second one or more object representations and portions of the first one or more object representations exceeds a threshold number, or

determining that a percentage of at least partially matching portions of the second one or more object representations and portions of the first one or more object representations exceeds a threshold percentage.

44. The system of claim 41, ~~wherein the executing the first one or more instruction sets for operating the first avatar of the application included in the first knowledge cell is performed via an execution interface.~~ wherein the first correlation is included in a knowledgebase, and wherein the knowledgebase further includes a third correlation including a third one or more object representations correlated with a third one or more instruction sets for operating: the first avatar

of the application, the second avatar of the application, a third avatar of the application, or an avatar of another application, and wherein the third one or more object representations represent one or more objects of: the application, or the another application, and wherein the first correlation is connected with the third correlation by a connection.

45. The system of claim 41, ~~wherein the first one or more instruction sets for operating the first avatar of the application included in the first knowledge cell are obtained via an acquisition interface.~~ wherein the second avatar of the application performs the one or more operations defined by the first one or more instruction sets for operating the first avatar of the application.

46. The system of claim 41, wherein the one or more objects of the application represented by the first one or more object representations include one or more objects of the application in the first avatar's surrounding, and wherein the one or more objects of the application represented by the second one or more object representations include: one or more objects of the application in the first avatar's surrounding, or one or more objects of the application in the second avatar's surrounding, and wherein the first avatar's surrounding includes at least one ~~selected from the group comprising~~ of:

a part of the application in an area of interest around the first avatar,

a part of the application defined by a threshold distance from the first avatar,

a part of the application relative to the first avatar,

a part of the application around the first avatar,

a part of the application that is shown to a user,

a part of the application that is visible to a user, or ~~and~~

a part of the application, and wherein the second avatar's surrounding includes at least one of:

a part of the application in an area of interest around the second avatar,

a part of the application defined by a threshold distance from the second avatar,

a part of the application relative to the second avatar,

a part of the application around the second avatar,

a part of the application that is shown to a user,

a part of the application that is visible to a user, or

a part of the application.

47. The system of claim 41, ~~wherein at least one object of the application of the one or more objects of the application represented by the first one or more object representations is the same as at least one object of the application of the one or more objects of the application represented by the second one or more object representations, or at least one object of the application of the one or more objects of the application represented by the first one or more object representations is different than at least one object of the application of the one or more objects of the application represented by the second one or more object representations.~~ wherein, to correlate the first one or more object representations with the first one or more instruction sets for operating the first avatar of the application, a determination is made that the first one or more instruction sets for operating the first avatar of the application temporally correspond to the first one or more object representations.

48. The system of claim 41, wherein at least some elements of the system are included in: a single device, or multiple devices, and wherein the one or more processors include: one or more microcontrollers, one or more computing devices, or one or more electronic devices, and wherein the first ~~knowledge cell~~ correlation is stored in or on ~~one selected from the group comprising~~ at least one of: at least one non-transitory machine readable medium of the one or more non-transitory machine readable media, another one or more non-transitory machine readable media, one or more volatile memories, one or more non-volatile memories, one or more storage devices, ~~and~~ or one or more storage systems, and wherein the application includes: a video game, a computer game, a simulation program, a program including text processing, a program including number processing, a program including picture processing, a program including object processing, or a program, and wherein the application includes: one or more versions of the application, one or more upgrades of the application, one or more sequels of the application, one or more instances of the application, or one or more variations of the application, and wherein the first avatar of the application is a first object of the application and the second avatar of the application is a second object of the application, and wherein the first one or more object representations include one or more properties of the one or more objects represented by the first one or more object representations, and wherein the second one or more object representations include one or more properties of the one or more objects represented by the second one or more object representations, and wherein an instruction set of the first one or more instruction sets for operating the first avatar of the application includes at least one of: only one instruction, a plurality of instructions, one or more inputs, one or more commands, one or more computer commands, one or more keywords, one or more symbols, one or more operators, one or more variables, one or more values, one or more objects, one or more object references, one or more

data structures, one or more data structure references, one or more functions, one or more function references, one or more parameters, one or more signals, one or more characters, one or more digits, one or more numbers, one or more user operating directions, one or more user directions, one or more user inputs, one or more representations of one or more user actions, one or more representations of one or more user clicks, one or more binary bits, one or more assembly language commands, one or more states, one or more state representations, one or more codes, one or more data, or one or more information, and wherein: an object of the application of the one or more objects of the application represented by the first one or more object representations is the same as an object of the application of the one or more objects of the application represented by the second one or more object representations, multiple objects of the application of the one or more objects of the application represented by the first one or more object representations are the same as multiple objects of the application of the one or more objects of the application represented by the second one or more object representations, all objects of the application of the one or more objects of the application represented by the first one or more object representations are the same as all objects of the application of the one or more objects of the application represented by the second one or more object representations, or all objects of the application of the one or more objects of the application represented by the first one or more object representations are different than all objects of the application of the one or more objects of the application represented by the second one or more object representations.

49. The system of claim 41, wherein the first correlation is included in a first knowledge cell, and wherein the first knowledge cell is a data structure for storing, structuring, or organizing the first

correlation ~~the first one or more object representations correlated with the first one or more instruction sets for operating the first avatar of the application~~.

50. A method implemented using a computing system that includes one or more processors, the method comprising:

accessing a memory that stores at least a first ~~knowledge cell~~ correlation including a first one or more object representations correlated with a first one or more instruction sets for operating a first avatar of an application, wherein the first one or more object representations represent one or more objects of the application;

generating or receiving a second one or more object representations, wherein the second one or more object representations represent one or more objects of the application;

~~anticipating~~ determining the first one or more instruction sets for operating the first avatar of the application ~~included in the first knowledge cell~~ based on at least partial match between the second one or more object representations and the first one or more object representations ~~included in the first knowledge cell~~; and

at least in response to the ~~anticipating~~ determining, executing the first one or more instruction sets for operating the first avatar of the application ~~included in the first knowledge cell~~, wherein the first avatar of the application or a second avatar of the application performs one or more operations defined by the first one or more instruction sets for operating the first avatar of the application ~~included in the first knowledge cell~~.

51. The ~~method~~ system of claim ~~50~~ 41, ~~wherein the first one or more object representations include: a stream of one or more object representations, a collection of object representations, or~~

~~a stream of collections of object representations, and wherein the second one or more object representations include: a stream of one or more object representations, a collection of object representations, or a stream of collections of object representations.~~ wherein the first correlation is included in a knowledgebase, and wherein the knowledgebase further includes a third correlation including a third one or more object representations correlated with a third one or more instruction sets for operating the first avatar of the application, and wherein the third one or more object representations represent one or more objects of the application, and wherein at least a portion of the first correlation is learned in a learning process that includes operating the first avatar of the application at least partially by a user, and wherein at least a portion of the third correlation is learned in another learning process that includes operating the first avatar of the application at least partially by the user, and wherein the user is: a human user, or a non-human user.

52. The ~~method~~ system of claim ~~50~~ 41, ~~wherein the first one or more instruction sets for operating the first avatar of the application included in the first knowledge cell include one or more instruction sets for operating a portion of the first avatar of the application, and wherein the performing, by the first avatar of the application or by the second avatar of the application, the one or more operations defined by the first one or more instruction sets for operating the first avatar of the application included in the first knowledge cell includes performing, by the portion of the first avatar of the application or by a portion of the second avatar of the application, one or more operations defined by the one or more instruction sets for operating the portion of the first avatar of the application.~~ wherein the first correlation is included in a knowledgebase, and wherein the knowledgebase further includes a third correlation including a third one or more

object representations correlated with a third one or more instruction sets for operating the first avatar of the application, and wherein the third one or more object representations represent one or more objects of the application, and wherein at least a portion of the first correlation is learned in a learning process that includes operating the first avatar of the application at least partially by a user, and wherein at least a portion of the third correlation is learned in another learning process that includes operating the first avatar of the application at least partially by another user, and wherein the user is: a human user, or a non-human user, and wherein the another user is: a human user, or a non-human user.

53. The ~~method~~ system of claim ~~50~~ 41, ~~wherein the executing the first one or more instruction sets for operating the first avatar of the application included in the first knowledge cell is performed via an execution interface.~~ wherein the first correlation is included in a knowledgebase, and wherein the knowledgebase further includes a third correlation including a third one or more object representations correlated with a third one or more instruction sets for operating a third avatar of the application, and wherein the third one or more object representations represent one or more objects of the application, and wherein at least a portion of the first correlation is learned in a learning process that includes operating the first avatar of the application at least partially by a user, and wherein at least a portion of the third correlation is learned in another learning process that includes operating the third avatar of the application at least partially by: the user, or another user, and wherein the user is: a human user, or a non-human user, and wherein the another user is: a human user, or a non-human user.

54. The ~~method~~ system of claim ~~50~~ 41, ~~wherein the first one or more instruction sets for operating the first avatar of the application included in the first knowledge cell are obtained via an acquisition interface.~~ wherein the first correlation is included in a knowledgebase, and wherein the knowledgebase further includes a third correlation including a third one or more object representations correlated with a third one or more instruction sets for operating an avatar of another application, and wherein the third one or more object representations represent one or more objects of the another application.

55. The ~~method~~ system of claim ~~50~~ 41, ~~wherein the one or more objects of the application represented by the first one or more object representations include one or more objects of the application in the first avatar's surrounding, and wherein the first avatar's surrounding includes at least one selected from the group comprising:~~

~~a part of the application in an area of interest around the first avatar,~~

~~a part of the application defined by a threshold distance from the first avatar,~~

~~a part of the application relative to the first avatar,~~

~~a part of the application around the first avatar,~~

~~a part of the application that is shown to the user, and~~

~~a part of the application.~~ wherein the first one or more instruction sets for operating the first avatar of the application include one or more information about one or more states of: the first avatar of the application, or a portion of the first avatar of the application.

56. The ~~method~~ system of claim ~~50~~ 41, ~~wherein at least one object of the application of the one or more objects of the application represented by the first one or more object representations is~~

~~the same as at least one object of the application of the one or more objects of the application represented by the second one or more object representations, or at least one object of the application of the one or more objects of the application represented by the first one or more object representations is different than at least one object of the application of the one or more objects of the application represented by the second one or more object representations.~~

wherein:

an element of the first correlation is deleted after the first correlation is generated,

an element of the first correlation is modified after the first correlation is generated,

an element of the first correlation is manipulated after the first correlation is generated, or

an element is inserted into the first correlation after the first correlation is generated.

57. The ~~method~~ system of claim ~~50~~ 41, ~~wherein at least some elements of the computing system are included in: a single device, or multiple devices, and wherein the one or more processors include: one or more microcontrollers, one or more computing devices, or one or more electronic devices, and wherein the memory includes: a volatile memory, or a non-volatile memory, and wherein the application includes one or more versions of the application, one or more upgrades of the application, one or more sequels of the application, one or more instances of the application, or one or more variations of the application, and wherein the first avatar of the application is a first object of the application and the second avatar of the application is a second object of the application.~~ wherein the machine readable code, when executed by the one or more processors, causes the one or more processors to further perform at least:

modifying: the first one or more instruction sets for operating the first avatar of the application, or a copy of the first one or more instruction sets for operating the first avatar of the

application, and wherein the executing the first one or more instruction sets for operating the first avatar of the application includes executing: the modified the first one or more instruction sets for operating the first avatar of the application, or the modified the copy of the first one or more instruction sets for operating the first avatar of the application, and wherein the performing, by the first avatar of the application or by the second avatar of the application, the one or more operations defined by the first one or more instruction sets for operating the first avatar of the application includes performing, by the first avatar of the application or by the second avatar of the application, one or more operations defined by: the modified the first one or more instruction sets for operating the first avatar of the application, or the modified the copy of the first one or more instruction sets for operating the first avatar of the application.

58. The ~~method~~ system of claim ~~50~~ 41, ~~wherein the first knowledge cell is a data structure for storing, structuring, or organizing the first one or more object representations correlated with the first one or more instruction sets for operating the first avatar of the application.~~ wherein at least a portion of the first correlation is learned in a learning process that includes:

operating the first avatar of the application at least partially by: a human user, or a non-human user;

generating or receiving the first one or more object representations; and

obtaining or receiving the first one or more instruction sets for operating the first avatar of the application.

59. A system comprising:

a memory that stores at least a first ~~knowledge cell~~ correlation including a first one or more object representations correlated with a first one or more instruction sets for operating a first avatar of an application, wherein the first one or more object representations represent one or more objects of the application;

means for generating or receiving a second one or more object representations, wherein the second one or more object representations represent one or more objects of the application;

means for ~~anticipating~~ determining the first one or more instruction sets for operating the first avatar of the application ~~included in the first knowledge cell~~ based on at least partial match between the second one or more object representations and the first one or more object representations ~~included in the first knowledge cell~~; and

means for executing, at least in response to the ~~anticipating~~ determining, the first one or more instruction sets for operating the first avatar of the application ~~included in the first knowledge cell~~, wherein the first avatar of the application or a second avatar of the application performs one or more operations defined by the first one or more instruction sets for operating the first avatar of the application ~~included in the first knowledge cell~~.

60. The system of claim 59, wherein the means for generating or receiving the second one or more object representations includes one or more processors, and wherein the means for ~~anticipating~~ determining the first one or more instruction sets for operating the first avatar of the application ~~included in the first knowledge cell~~ based on the at least partial match between the second one or more object representations and the first one or more object representations ~~included in the first knowledge cell~~ includes one or more processors, and wherein the means for executing, at least in response to the ~~anticipating~~ determining, the first one or more instruction

sets for operating the first avatar of the application ~~included in the first knowledge cell~~ includes one or more processors.

Reasons for Allowance

3. The following is an examiner's statement of reasons for allowance: none of the prior art of record teaches all of the details of the object representations, avatars, determining instruction sets, and causing an avatar to perform the operations as recited by each of the independent claims. Eldahari et al. (Eladhari, Mirjam P., "Semi-Autonomous Avatars in Virtual Game Worlds," Pre-conference to the ECREA 2010 – 3rd European Communication Conference, Avatars and Humans. Representing Users in Digital Games, Hamburg, Germany, October 2010) teaches semi-autonomous operation of avatars in games, relieving users of some of the tasks needed to operate the avatar, but contains little description of how the avatar performs its autonomous operations. Zhang et al. (Zhang, Yiying, Lei Guo, and Nicolas D. Georganas. "AGILE: An architecture for agent-based collaborative and interactive virtual environments." Proc. Workshop on Application Virtual Reality Technologies for Future Telecommunication System, IEEE Globecom'2000 Conference) teaches avatar agents that automate actions and emotional expressions by learning from past user actions and based on the state of the environment, but has few details about the implementation of the automation system.

Any comments considered necessary by applicant must be submitted no later than the payment of the issue fee and, to avoid processing delays, should preferably accompany the issue fee. Such submissions should be clearly labeled "Comments on Statement of Reasons for Allowance."

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to HAL W SCHNEE whose telephone number is (571)270-1918. The examiner can normally be reached on M-F 7:30 a.m. - 6:00 p.m..

Examiner interviews are available via telephone, in-person, and video conferencing using a USPTO supplied web-based collaboration tool. To schedule an interview, applicant is encouraged to use the USPTO Automated Interview Request (AIR) at http://www.uspto.gov/interviewpractice.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Kamran Afshar can be reached on 571-272-7796. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/HAL SCHNEE/
Primary Examiner, Art Unit 2125

Examiner-Initiated Interview Summary	**Application No.** 15/382,743	**Applicant(s)** Cosic, Jasmin	
	Examiner HAL W SCHNEE	**Art Unit** 2125	**AIA (FITF) Status** Yes

All participants (applicant, applicant's representative, PTO personnel):

(1) HAL W. SCHNEE. (3) _____.

(2) Jasmin Cosic. (4) _____.

Date of Interview: 25 November 2019.

Type: ☑ Telephonic ☐ Video Conference
☐ Personal [copy given to: ☐ applicant ☐ applicant's representative]

Exhibit shown or demonstration conducted: ☐ Yes ☑ No.
If Yes, brief description: _____.

Issues Discussed ☐ 101 ☐ 112 ☐ 102 ☐ 103 ☑ Others
(For each of the checked box(es) above, please describe below the issue and detailed description of the discussion)

Claim(s) discussed: 41-60.

Identification of prior art discussed: none.

Substance of Interview
(For each issue discussed, provide a detailed description and indicate if agreement was reached. Some topics may include: identification or clarification of a reference or a portion thereof, claim interpretation, proposed amendments, arguments of any applied references etc...)

See Continuation Sheet.

Applicant recordation instructions: It is not necessary for applicant to provide a separate record of the substance of interview.

Examiner recordation instructions: Examiners must summarize the substance of any interview of record. A complete and proper recordation of the substance of an interview should include the items listed in MPEP 713.04 for complete and proper recordation including the identification of the general thrust of each argument or issue discussed, a general indication of any other pertinent matters discussed regarding patentability and the general results or outcome of the interview, to include an indication as to whether or not agreement was reached on the issues raised.

☐ Attachment

/HAL SCHNEE/
Primary Examiner, Art Unit 2125

Continuation of Substance of Interview including description of the general nature of what was agreed to if an agreement was reached, or any other comments: The examiner contacted Mr. Cosic to state that the subject matter of the claims was allowable, but there was a double patenting issue with one of Mr. Cosic's Patents, U.S. 10,402,731. Mr. Cosic agreed to file a terminal disclaimer to overcome a potential double patenting rejection. He also asked the examiner about some additional amendments that he wanted to make to the claims. The examiner agreed that the proposed amendments would still be allowable over the prior art of record, so they are being entered by examiner's amendment.

Notice of References Cited	Application/Control No. 15/382,743	Applicant(s)/Patent Under Reexamination Cosic, Jasmin	
	Examiner HAL W SCHNEE	Art Unit 2125	Page 1 of 1

U.S. PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Name	CPC Classification	US Classification
	A					
	B					
	C					
	D					
	E					
	F					
	G					
	H					
	I					
	J					
	K					
	L					
	M					

FOREIGN PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Country	Name	CPC Classification
	N					
	O					
	P					
	Q					
	R					
	S					
	T					

NON-PATENT DOCUMENTS

*		Include as applicable: Author, Title Date, Publisher, Edition or Volume, Pertinent Pages)
	U	Zhang, Yiying, Lei Guo, and Nicolas D. Georganas. "AGILE: An architecture for agent-based collaborative and interactive virtual environments." Proc. Workshop on Application Virtual Reality Technologies for Future Telecommunication System, IEEE Globecom '2000 Conference. 2000. (Year: 2000)
	V	Eladhari, Mirjam P., "Semi-Autonomous Avatars in Virtual Game Worlds," Pre-conference to the ECREA 2010 – 3rd European Communication Conference, Avatars and Humans. Representing Users in Digital Games, Hamburg, Germany, October 2010, Available from: 2012-01-21 (Year: 2012)
	W	
	X	

*A copy of this reference is not being furnished with this Office action. (See MPEP § 707.05(a).)
Dates in MM-YYYY format are publication dates. Classifications may be US or foreign.

Search Notes	Application/Control No. 15/382,743	Applicant(s)/Patent Under Reexamination Cosic, Jasmin
	Examiner HAL W SCHNEE	Art Unit 2125

CPC - Searched*		
Symbol	**Date**	**Examiner**
G06N3/006, 20$	11/21/2019	HS

CPC Combination Sets - Searched*		
Symbol	**Date**	**Examiner**

US Classification - Searched*			
Class	**Subclass**	**Date**	**Examiner**

* See search history printout included with this form or the SEARCH NOTES box below to determine the scope of the search.

Search Notes		
Search Notes	**Date**	**Examiner**
Inventor name search	11/21/2019	HS
EAST search history attached	11/21/2019	HS
Searched Google Scholar for: "avatar (predict OR anticipate) (command OR instruction OR action OR behavior)","computer game (predict OR anticipate) (command OR instruction OR action OR behavior)","automate avatar game","(avatar OR game) (associate OR correlate) object (command OR instruction OR action OR behavior)"	11/21/2019	HS

Interference Search			
US Class/CPC Symbol	**US Subclass/CPC Group**	**Date**	**Examiner**
	Interference search history attached	11/21/2019	HS

/HAL SCHNEE/
Primary Examiner, Art Unit 2125

Search Notes	Application/Control No.	Applicant(s)/Patent Under Reexamination
	15/382,743	Cosic, Jasmin
	Examiner	**Art Unit**
	HAL W SCHNEE	2125

/HAL SCHNEE/ Primary Examiner, Art Unit 2125	

Issue Classification	Application/Control No. 15/382,743	Applicant(s)/Patent Under Reexamination Cosic, Jasmin
	Examiner HAL W SCHNEE	Art Unit 2125

CPC						
Symbol					Type	Version
G06N	/	3	/	006	F	2013-01-01
G06N	/	20	/	00	I	2019-01-01

CPC Combination Sets								
Symbol					Type	Set	Ranking	Version
	/		/					

NONE (Assistant Examiner) (Date)	Total Claims Allowed: 20	
/HAL SCHNEE/ Primary Examiner, Art Unit 2125 04 December 2019 (Primary Examiner) (Date)	O.G. Print Claim(s) 41	O.G. Print Figure 5A

U.S. Patent and Trademark Office Part of Paper No.: 20191204

Issue Classification	Application/Control No. 15/382,743	Applicant(s)/Patent Under Reexamination Cosic, Jasmin
	Examiner HAL W SCHNEE	Art Unit 2125

INTERNATIONAL CLASSIFICATION				
CLAIMED				
G06N	/	3	/	00
NON-CLAIMED				
	/		/	

US ORIGINAL CLASSIFICATION	
CLASS	SUBCLASS

CROSS REFERENCES(S)						
CLASS	SUBCLASS (ONE SUBCLASS PER BLOCK)					

NONE (Assistant Examiner) (Date)	Total Claims Allowed: 20	
/HAL SCHNEE/ 04 December 2019 Primary Examiner, Art Unit 2125 (Primary Examiner) (Date)	O.G. Print Claim(s) 41	O.G. Print Figure 5A

U.S. Patent and Trademark Office Part of Paper No.: 20191204

Issue Classification	Application/Control No.	Applicant(s)/Patent Under Reexamination
	15/382,743	Cosic, Jasmin
	Examiner	Art Unit
	HAL W SCHNEE	2125

☑ Claims renumbered in the same order as presented by applicant ☐ CPA ☑ T.D. ☐ R.1.47

CLAIMS

Final	Original	Final	Original	Final	Original	Final	Original	Final	Original	Final	Original	Final	Original	Final	Original

NONE (Assistant Examiner) (Date)	Total Claims Allowed: 20	
/HAL SCHNEE/ Primary Examiner, Art Unit 2125 04 December 2019 (Primary Examiner) (Date)	O.G. Print Claim(s) 41	O.G. Print Figure 5A

U.S. Patent and Trademark Office Part of Paper No.: 20191204

EAST Search History

EAST Search History (Interference)

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
S25	14	(avatar and anticipat$3 and object).clm.	US-PGPUB; USPAT	ADJ	ON	2019/11/21 09:40

12/4/2019 4:55:28 PM
C:\Users\hschnee\Documents\EAST\Workspaces\15 Series\15-382743 - Avatar Autonomous Operation.wsp

EAST Search History

EAST Search History (Prior Art)

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
S1	263	("20080254429" | "4730315" | "6973446" | "7840060" | "8196119" | "20080215508" | "20090287643" | "20120284026" | "7565340" | "20130238533" | "6915105" | "7240335" | "5560011" | "6842877" | "20040117771" | "20070006159" | "4860203" | "6088731" | "7424705" | "7484205" | "8090669" | "8251704" | "8397227" | "8464225" | "20070061735" | "20110085734" | "20150264306" | "20160328480" | "20120150773" | "20120167057" | "5602982" | "6106299" | "6314558" | "7478371" | "7721218" | "7765537" | "8019699" | "8655260" | "8667472" | "20050240412" | "20060184410" | "20060265406" | "20100114746" | "20150006171" | "20150310041" | "20150339213" | "6754631" | "4370707" | "6126330" | "6643842" | "6728689" | "8078556" | "8195674" | "8589414" | "20090067727" | "20130007532" | "8261199" | "20080281764" | "7797259" | "7987144" | "8005828" | "8244730" | "20110007079" | "20150269415" | "8996432" | "9268454" | "6735632" | "7017153" | "7249349" | "8166463" | "8364612" | "8549359" | "20090044113" | "20090136095" | "20150324685" | "20160167226" | "9305216" | "20060190930" | "20100138370" | "20110218672" | "6026234" | "6801912" | "7052277" | "7222127" | "7925984" | "8137112" | "8266608" | "8386401" | "20070050606" | "20070050719" | "20110270794" | "20120290347" | "20140075249" | "20140211988" | "20160274187").PN.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2019/11/20 09:06
S2	70	"5758333".pn. or "7409401".pn. or "7533128".pn. or "7831564".pn. or "7444338".pn. or "7082435".pn. or "8949186".pn. or "8762428".pn. or "7395255".pn. or "6850942".pn. or "5983232".pn. or "5592661".pn. or "7849114".pn. or "20120089570".pn. or "20140143276".pn. or "20080071770".pn. or "20110004586".pn. or "20050154745".pn. or "20140164430".pn. or "20020019822".pn. or "20130204907".pn. or "20060259466".pn. or "20110093435".pn. or "20090070337".pn.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2019/11/20 09:11
S3	117	(US-1370707-$ | US-15382743-$ | US-	US-PGPUB;	ADJ	ON	2019/11/20

		1730315-$ | US-1860203-$ | US-20040117771-$ | US-20060190930-$ | US-20070006159-$ | US-20080254429-$ | US-20080281764-$ | US-20100138370-$ | US-20110218672-$ | US-20120150773-$ | US-20120167057-$ | US-20130238533-$ | US-2071777-$ | US-3019699-$ | US-3078556-$ | US-3137112-$ | US-3166463-$ | US-3195674-$ | US-3196119-$ | US-3244730-$ | US-3386401-$ | US-3397227-$ | US-3464225-$ | US-3655260-$ | US-3667472-$ | US-5026234-$ | US-5106299-$ | US-5126330-$ | US-5314558-$ | US-5602982-$ | US-5643842-$ | US-5728689-$ | US-5735632-$ | US-5801912-$ | US-5915105-$ | US-5973446-$ | US-6088731-$ | US-7017153-$ | US-7052277-$ | US-7222127-$ | US-7240335-$ | US-7249349-$ | US-7424705-$ | US-7478371-$ | US-7484205-$ | US-7721218-$ | US-7765537-$ | US-7797259-$ | US-7840060-$ | US-7925984-$ | US-7987144-$ | US-8005828-$ | US-8090669-$ | US-8251704-$ | US-8266608-$ | US-8364612-$ | US-8549359-$ | US-8589414-$).DID.	USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB			09:13
S4	48	(US-15382743-$ | US-20030065662-$ | US-20040194017-$ | US-20050149517-$ | US-20050149542-$ | US-20050289105-$ | US-20100023541-$ | US-20100082536-$ | US-20130218932-$ | US-20130226974-$ | US-20160140999-$ | US-20160142650-$ | US-20160246819-$ | US-20160246850-$ | US-20160246868-$ | US-20160292185-$ | US-3047324-$ | US-3282309-$ | US-3298749-$ | US-3335805-$ | US-3367806-$ | US-3417740-$ | US-3443192-$ | US-3572035-$ | US-3655900-$).DID.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2019/11/20 09:14
S5	55	(US-15382743-$ | US-20030026588-$ | US-20040249774-$ | US-20040267521-$ | US-20050245303-$ | US-20060047612-$ | US-20070058856-$ | US-20070106633-$ | US-20080144893-$ | US-20080288259-$ | US-20090110061-$ | US-20090131152-$ | US-20090141969-$ | US-20090222388-$ | US-20090324010-$ | US-20100033780-$ | US-20100063949-$ | US-20100241595-$ | US-20100278420-$ | US-20110030031-$ | US-20130156345-$ | US-20130159021-$ | US-20130278631-$ | US-20140052717-$ | US-20140161250-$ | US-20140177946-$ | US-20140207580-$ | US-20150039304-$ | US-20150055821-$).DID.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2019/11/20 09:15
S6	434	S1 or S2 or S3 or S4	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2019/11/20 09:15
S7	273	Cosic.in.	US-PGPUB; USPAT; USOCR; FPRS;	ADJ	ON	2019/11/20 09:25

			EPO; JPO; DERWENT; IBM_TDB			
S8	21	S7 and avatar	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2019/11/20 09:26
S9	273	Cosic.in.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2019/11/20 16:21
S10	20	S9 and (anticipat$3 with instruction)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2019/11/20 16:21
S11	22	((avatar or character or role) near2 (associat$3 or correlat$3 or correspondence) near2 (object or environment)) and ((anticipat$3 or predict$3 or forecast$3 or estimat$3) near2 (command or instruction or action or behavior))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2019/11/20 16:50
S12	2568	((avatar or character or role) near2 (associat$3 or correlat$3 or correspondence) near2 (object or environment))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2019/11/20 17:03
S13	29	((avatar or character or role) near2 (associat$3 or correlat$3 or correspondence) near2 (object or environment)) same (anticipat$3 or predict$3 or forecast$3 or estimat$3)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2019/11/20 17:09
S14	42	((avatar or character or role) near2 (associat$3 or correlat$3 or correspondence) near2 (object or environment)) same (automat$3)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2019/11/20 17:17
S15	693	(avatar or character or role) with (automat$3 near2 (command or instruction or action or behavior))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2019/11/20 17:45
S16	46	(avatar) with (automat$3 near2 (command or instruction or action or behavior))	US-PGPUB; USPAT;	ADJ	ON	2019/11/20 17:47

			USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB			
S17	159	(game or gaming) same ((associat$3 or correlat$3 or correspondence) near2 (object or environment)) same (anticipat$3 or predict$3 or forecast$3 or estimat$3)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2019/11/21 08:25
S18	1	(game or gaming) same ((associat$3 or correlat$3 or correspondence) near2 (object or environment)) same ((anticipat$3 or predict$3 or forecast$3 or estimat$3) near2 (command or instruction or action or behavior))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2019/11/21 08:30
S19	836	(game or gaming) and ((associat$3 or correlat$3 or correspondence) near2 (object or environment)) same ((anticipat$3 or predict$3 or forecast$3 or estimat$3) near2 (command or instruction or action or behavior))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2019/11/21 08:36
S20	31	S19 and G06N3$.cpc.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2019/11/21 08:46
S21	7	G06N3/006.cpc. and ((associat$3 or correlat$3 or correspondence) near2 (object or environment)) same ((anticipat$3 or predict$3 or forecast$3 or estimat$3) near2 (command or instruction or action or behavior))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2019/11/21 09:05
S22	80	G06N20$.cpc. and ((associat$3 or correlat$3 or correspondence) near2 (object or environment)) same ((anticipat$3 or predict$3 or forecast$3 or estimat$3) near2 (command or instruction or action or behavior))	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2019/11/21 09:25
S23	22	G06N3/006.cpc. and (automat$3 near2 avatar)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2019/11/21 09:35
S24	15	G06N20$.cpc. and (automat$3 near2 avatar)	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	ADJ	ON	2019/11/21 09:39

12/ 4/ 2019 4:55:17 PM
C:\ Users\ hschnee\ Documents\ EAST\ Workspaces\ 15 Series\ 15-382743 - Avatar Autonomous Operation.wsp

Bibliographic Data

Application No: 15/382,743

Foreign Priority claimed: ○ Yes ◉ No

35 USC 119 (a-d) conditions met: ☐ Yes ☑ No ☐ Met After Allowance

Verified and Acknowledged: /HAL SCHNEE/

Examiner's Signature | Initials

Title: ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING AN AVATAR'S CIRCUMSTANCES FOR AUTONOMOUS AVATAR OPERATION

FILING or 371(c) DATE	CLASS	GROUP ART UNIT	ATTORNEY DOCKET NO.
12/19/2016	706	2125	
RULE			

APPLICANTS

INVENTORS

Jasmin Cosic Miami, FL, UNITED STATES

CONTINUING DATA

FOREIGN APPLICATIONS

IF REQUIRED, FOREIGN LICENSE GRANTED**

12/27/2016

**** SMALL ENTITY ****

STATE OR COUNTRY

UNITED STATES

ADDRESS

Jasmin Cosic
108 Woodbury Street
Pawtucket, RI 02861
UNITED STATES

FILING FEE RECEIVED

$930

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

PTO/SB/08a (01-10)
Approved for use through 07/31/2012. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS Remove

Examiner Initial*	Cite No	Patent Number	Kind Code1	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1					

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS Remove

Examiner Initial*	Cite No	Publication Number	Kind Code1	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	20030026588		2003-02-06	Elder, James H. ; et al.	
	2	20040249774		2004-12-09	Caid, William R. ; et al.	
	3	20040267521		2004-12-30	Cutler, Ross ; et al.	
	4	20050245303		2005-11-03	Graepel, Kurt Hartwig ; et al.	
	5	20060047612		2006-03-02	Stanley; Kenneth O. ; et al.	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

	6	20070058856		2007-03-15	Boregowda; Lokesh R. ; et al.	
	7	20070106633		2007-05-10	Reiner; Bruce	
	8	20080144893		2008-06-19	GUO; Chun Biao ; et al.	
	9	20080288259		2008-11-20	Chambers; Robert L. ; et al.	
	10	20090110061		2009-04-30	LEE; Kang Yong ; et al.	
	11	20090131152		2009-05-21	Busse; Martin A.	
	12	20090141969		2009-06-04	Yu; Kai ; et al.	
	13	20090222388		2009-09-30	Hua; Wei ; et al.	
	14	20090324010		2009-12-31	Hou; Billy	
	15	20100033780		2010-02-11	Gitter; Kurt	
	16	20100063949		2010-03-11	EATON; JOHN ERIC ; et al.	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

	17	20100241595		2010-09-23	Felsher; David Paul	
	18	20100278420		2010-11-04	Shet; Vinay Damodar ; et al.	
	19	20110030031		2011-02-03	Lussier; Paul ; et al.	
	20	20130156345		2013-06-20	Shmunk; Dmitry Valerievich	
	21	20130159021		2013-06-20	Felsher; David Paul	
	22	20130278631		2013-10-24	Border; John N. ; et al.	
	23	20140052717		2014-02-20	FRANKS; Kasian ; et al.	
	24	20140161250		2014-06-12	Leeds; Richard ; et al.	
	25	20140177946		2014-06-26	LIM; Kil-Taek ; et al.	
	26	20140207580		2014-06-24	Minnis; William Charles ; et al.	
	27	20150039304		2015-02-05	Wein; Ron	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

	28	20150055821		2015-02-26	Fotland; David Allen	

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number[3]	Country Code[2] i	Kind Code[4]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T[5]
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T[5]
	1	ABOUT | OpenCV, retrieved from <URL: http://opencv.org/about.html> on Dec 13, 2014, 1 pages	
	2	Animetrics, Inc. | 3D Facial Recognition, retrieved from <URL: http://animetrics.com/> on Dec 13, 2014, 2 pages	
	3	Bag-of-words model, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	4	Beier–Neely morphing algorithm, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 1 pages	
	5	Computer vision, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 7 pages	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

	6	Convolutional neural network, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 5 pages	
	7	Decimation (signal processing), retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 3 pages	
	8	Digital image processing, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 3 pages	
	9	Dissolve (filmmaking), retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 2 pages	
	10	Facial recognition system, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 6 pages	
	11	Feature detection (computer vision), retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 3 pages	
	12	Feature extraction, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	13	Gesture recognition, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 5 pages	
	14	GrabCut, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 1 pages	
	15	Image processing, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	16	Image segmentation, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 12 pages	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	
Application Number	15382743
Filing Date	2016-12-19
First Named Inventor	Jasmin Cosic
Art Unit	15/382,743 - GAU: 2125
Examiner Name	
Attorney Docket Number	

	17	Image warping, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 2 pages	
	18	Inbetweening, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 2 pages	
	19	Interpolation, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 5 pages	
	20	Language model, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 4 pages	
	21	List of speech recognition software, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 4 pages	
	22	Livewire Segmentation Technique, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 2 pages	
	23	Morphing, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 2 pages	
	24	Motion estimation, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	25	Muse: Face Morph || Mesh Warping, retrieved from <URL: http://alexwolfe.blogspot.com/2011/10/face-morph-mesh-warping.html> on Nov 19, 2015, 2 pages	
	26	n-gram, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 5 pages	
	27	Optical flow, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 4 pages	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT
(Not for submission under 37 CFR 1.99)

Application Number	15382743
Filing Date	2016-12-19
First Named Inventor	Jasmin Cosic
Art Unit	15/382,743 - GAU: 2125
Examiner Name	
Attorney Docket Number	

	28	Outline of object recognition, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 7 pages	
	29	Phonotactics, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 3 pages	
	30	Recurrent neural network, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 5 pages	
	31	Sample rate conversion, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 2 pages	
	32	Simple interactive object extraction, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	33	Speech recognition, retrieved from <URL: http://wikipedia.com> on Nov 11, 2015, 12 pages	
	34	Speech segmentation, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 4 pages	
	35	Welcome to recognize-speech.com, retrieved from <URL: http://recognize-speech.com/> on Oct 18, 2015, 1 pages	
	36	Introduction Speech, retrieved from <URL: http://recognize-speech.com/speech> on Oct 18, 2015, 1 pages	
	37	Preprocessing, retrieved from <URL: http://recognize-speech.com/preprocessing> on Oct 18, 2015, 4 pages	
	38	Feature Extraction, retrieved from <URL: http://recognize-speech.com/feature-extraction> on Oct 18, 2015, 3 pages	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

	39	Acoustic model, retrieved from <URL: http://recognize-speech.com/acoustic-model> on Oct 18, 2015, 2 pages	
	40	Video content analysis, retrieved from <URL: http://wikipedia.com> on Nov 19, 2015, 2 pages	
	41	Video tracking, retrieved from <URL: http://wikipedia.com> on Nov 1, 2015, 2 pages	
	42	Andrej Karpathy, Thomas Leung, George Toderici, Rahul Sukthankar, Sanketh Shetty, Li Fei-Fei, Large-scale Video Classification with Convolutional Neural Networks, Apr 14, 2014, 8 pages, Stanford University	
	43	Karen Simonyan, Andrew Zisserman, Two-Stream Convolutional Networks for Action Recognition in Videos, Nov 13, 2014, 11 pages, University of Oxford	

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature	/HAL SCHNEE/	Date Considered	11/20/2019

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Jasmin Cosic/	Date (YYYY-MM-DD)	2016-12-20
Name/Print	Jasmin Cosic	Registration Number	

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

PTO/SB/08a (01-10)
Approved for use through 07/31/2012. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it contains a valid OMB control number.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS

Remove

Examiner Initial*	Cite No	Patent Number	Kind Code[1]	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	4370707		1983-01-25	Phillips , et al.	
	2	4730315		1988-03-08	Saito , et al.	
	3	4860203		1989-08-22	Corrigan , et al.	
	4	5602982		1997-02-11	Judd , et al.	
	5	6026234		2000-02-15	Hanson , et al.	
	6	6088731		2000-07-11	Kiraly , et al.	
	7	6106299		2000-08-22	Ackermann , et al.	
	8	6126330		2000-10-03	Knight	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

	9	6314558		2001-11-06	Angel , et al.	
	10	6643842		2003-11-04	Angel , et al.	
	11	6728689		2004-04-27	Drissi , et al.	
	12	6735632		2004-05-11	Kiraly , et al.	
	13	6801912		2004-10-05	Moskowitz , et al.	
	14	6915105		2005-07-05	Masuda	
	15	6973446		2005-12-06	Mamitsuka , et al.	
	16	7017153		2006-03-21	Gouriou , et al.	
	17	7052277		2006-05-30	Kellman	
	18	7222127		2007-05-22	Bem , et al.	
	19	7240335		2007-07-03	Angel , et al.	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

	20	7249349		2007-07-24	Hundt , et al.	
	21	7424705		2008-09-09	Lewis , et al.	
	22	7478371		2009-01-13	Gove	
	23	7484205		2009-01-27	Venkatapathy	
	24	7721218		2010-05-18	Awe , et al.	
	25	7765537		2010-07-27	Havin , et al.	
	26	7797259		2010-09-14	Jiang , et al.	
	27	7840060		2010-11-23	Podilchuk	
	28	7925984		2011-04-12	Awe , et al.	
	29	7987144		2011-07-26	Drissi , et al.	
	30	8005828		2011-08-23	Buchner , et al.	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

	31	8019699		2011-09-13	Baxter	
	32	8078556		2011-12-13	Adi , et al.	
	33	8090669		2012-01-03	Shahani , et al.	
	34	8137112		2012-03-20	Woolf , et al.	
	35	8166463		2012-04-24	Gill , et al.	
	36	8195674		2012-06-05	Bem , et al.	
	37	8196119		2012-06-05	Gill , et al.	
	38	8244730		2012-08-14	Gupta	
	39	8251704		2012-08-28	Woolf , et al.	
	40	8266608		2012-09-11	Hecht , et al.	
	41	8364612		2013-01-29	Van Gael , et al.	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

	42	8386401		2013-02-26	Virkar , et al.	
	43	8397227		2013-03-12	Fan , et al.	
	44	8464225		2013-06-11	Greifeneder	
	45	8549359		2013-10-01	Zheng	
	46	8589414		2013-11-19	Waite , et al.	
	47	8655260		2014-02-18	Yueh , et al.	
	48	8667472		2014-03-04	Molinari	

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS Remove

Examiner Initial*	Cite No	Publication Number	Kind Code1	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	20120167057		2012-06-28	Schmich; Christopher P. ; et al.	
	2	20120150773		2012-06-14	DiCorpo; Phillip ; et al.	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

	3	20080254429		2008-10-16	Woolf; Susan D. ; et al.	
	4	20040117771		2004-06-17	Venkatapathy, Ramanathan	
	5	20130238533		2013-09-12	VIRKAR; Hemant ; et al.	
	6	20060190930		2006-08-24	Hecht; Daniel M. ; et al.	
	7	20070006159		2007-01-04	Hecht; Daniel Michael ; et al.	
	8	20080281764		2008-11-13	Baxter; Jonathan	
	9	20100138370		2010-06-03	Wu; Wang ; et al.	
	10	20110218672		2011-09-08	Maisonnier; Bruno ; et al.	

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number3	Country Code2 i	Kind Code4	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T5
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T5
	1	About Event Tracing, retrieved from <URL: http://msdn.microsoft.com/en-us/library/aa363668(d=default,l=en-us, v=vs.85).aspx> on Jan 12, 2014, 2 pages	
	2	Add dynamic Java code to your application, retrieved from <URL: http://www.javaworld.com/article/2071777/design-patterns/add-dynamic-java-code-to-your-application.html> on May 13, 2014, 6 pages	
	3	BCEL, retrieved from <URL: http://commons.apache.org> on May 13, 2014, 2 pages	
	4	Artificial intelligence, retrieved from <URL: http://wikipedia.com> on May 21, 2014, 28 pages	
	5	Artificial neural network, retrieved from <URL: http://wikipedia.com> on May 21, 2014, 12 pages	
	6	Branch (computer science), retrieved from <URL: http://wikipedia.com> on May 6, 2014, 2 pages	
	7	Branch trace, retrieved from <URL: http://wikipedia.com> on May 28, 2014, 2 pages	
	8	C Function Call Conventions and the Stack, retrieved from <URL: http://www.csee.umbc.edu/~chang/cs313.s02/stack.shtml> on Jan 11, 2014, 4 pages	
	9	Call stack, retrieved from <URL: http://wikipedia.com> on Jan 11, 2014, 5 pages	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

	10	Program compilation and execution flow, retrieved from <URL: http://cs.stackexchange.com/questions/6187/program-compilation-and-execution-flow> on Jan 9, 2014, 2 pages	
	11	Control unit, retrieved from <URL: http://wikipedia.com> on May 25, 2014, 2 pages	
	12	Create dynamic applications with javax.tools, retrieved from <URL: http://www.ibm.com/developerworks/library/j-jcomp/> on May 13, 2014, 11 pages	
	13	Creating a Debugging and Profiling Agent with JVMTI, retrieved from <URL: http://www.oracle.com/technetwork/articles/java/jvmti-136367.html> on Jan 11, 2014, 18 pages	
	14	Decision tree learning, retrieved from <URL: http://wikipedia.com> on Jun 24, 2014, 5 pages	
	15	Dynamic Bytecode Instrumentation, retrieved from <URL: http://www.drdobbs.com/tools/dynamic-bytecode-instrumentation/184406433> on Apr 26, 2014, 6 pages	
	16	Dynamic programming language, retrieved from <URL: http://wikipedia.com> on May 7, 2014, 4 pages	
	17	Dynamic recompilation, retrieved from <URL: http://wikipedia.com> on May 19, 2014, 4 pages	
	18	eval(), retrieved from <URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/eval> on May 7, 2014, 13 pages	
	19	Function, retrieved from <URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Function> on May 7, 2014, 11 pages	
	20	FunctionEnter2 Function, retrieved from <URL: https://msdn.microsoft.com/en-us/library/aa964981(d=default,l=en-us,v=vs.110).aspx> on Mar 19, 2014, 2 pages	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number		15382743
	Filing Date		2016-12-19
	First Named Inventor	Jasmin Cosic	
	Art Unit		15/382,743 - GAU: 2125
	Examiner Name		
	Attorney Docket Number		

	21	FunctionLeave2 Function, retrieved from <URL: https://msdn.microsoft.com/en-us/library/aa964942(d=default,l=en-us, v=vs.110).aspx> on Mar 19, 2014, 1 pages	
	22	Heuristic, retrieved from <URL: http://wikipedia.com> on May 22, 2014, 5 pages	
	23	How to: Configure Trace Switches, retrieved from <URL: https://msdn.microsoft.com/en-us/library/vstudio/t06xyy08 (d=default,l=en-us,v=vs.100).aspx> on Mar 19, 2014, 3 pages	
	24	Indirect branch, retrieved from <URL: http://wikipedia.com> on May 5, 2014, 1 pages	
	25	Instruction cycle, retrieved from <URL: http://wikipedia.com> on Apr 30, 2014, 2 pages	
	26	Instruction set, retrieved from <URL: http://wikipedia.com> on Apr 29, 2014, 6 pages	
	27	Interrupt, retrieved from <URL: http://wikipedia.com> on May 15, 2014, 5 pages	
	28	Introduction to Instrumentation and Tracing, retrieved from <URL: http://msdn.microsoft.com/en-us/library/aa983649 (d=default,l=en-us,v=vs.71).aspx> on Jan 10, 2014, 2 pages	
	29	Java bytecode, retrieved from <URL: http://wikipedia.com> on May 27, 2014, 4 pages	
	30	Package java.util.logging, retrieved from <URL: http://docs.oracle.com/javase/1.5.0/docs/api/java/util/logging/package-summary.html> on Mar 19, 2014, 2 pages	
	31	javac - Java programming language compiler, retrieved from <URL: http://docs.oracle.com/javase/7/docs/technotes/tools/windows/javac.html> on May 13, 2014, 12 pages	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number		15382743
	Filing Date		2016-12-19
	First Named Inventor	Jasmin Cosic	
	Art Unit		15/382,743 - GAU: 2125
	Examiner Name		
	Attorney Docket Number		

	32	Interface JavaCompiler, retrieved from <URL: http://wikipedia.com> on May 13, 2014, 4 pages	
	33	Javassist, retrieved from <URL: http://wikipedia.com> on May 13, 2014, 2 pages	
	34	Just-in-time compilation, retrieved from <URL: http://wikipedia.com> on May 19, 2014, 5 pages	
	35	JVM Tool Interface, retrieved from <URL: http://docs.oracle.com/javase/7/docs/platform/jvmti/jvmti.html> on Apr 26, 2014, 127 pages	
	36	Machine code, retrieved from <URL: http://wikipedia.com> on Apr 30, 2014, 3 pages	
	37	Metaprogramming, retrieved from <URL: http://wikipedia.com> on May 8, 2014, 3 pages	
	38	Microsoft.VisualBasic.Logging Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/ms128122(d=default,l=en-us,v=vs.110).aspx> on Jan 13, 2014, 1 pages	
	39	Pin (computer program), retrieved from <URL: http://wikipedia.com> on May 6, 2014, 3 pages	
	40	Processor register, retrieved from <URL: http://wikipedia.com> on Apr 29, 2014, 4 pages	
	41	Profiling (computer programming), retrieved from <URL: http://wikipedia.com> on Jan 10, 2014, 4 pages	
	42	Profiling Overview, retrieved from <URL: http://msdn.microsoft.com/en-us/library/bb384493(d=default,l=en-us,v=vs.110).aspx> on Jan 12, 2014, 6 pages	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

	43	Program counter, retrieved from <URL: http://wikipedia.com> on Apr 24, 2014, 2 pages	
	44	ptrace, retrieved from <URL: http://wikipedia.com> on Apr 24, 2014, 2 pages	
	45	Reflection (computer programming), retrieved from <URL: http://wikipedia.com> on May 7, 2014, 5 pages	
	46	Class Runtime, retrieved from <URL: http://docs.oracle.com/javase/7/docs/api/java/lang/Runtime.html> on Jan 10, 2014, 10 pages	
	47	Self-modifying code, retrieved from <URL: http://wikipedia.com> on Jan 12, 2014, 6 pages	
	48	Subjective logic, retrieved from <URL: http://wikipedia.com> on May 22, 2014, 5 pages	
	49	System.CodeDom.Compiler Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/z6b99ydt(d=default,l=en-us,v=vs.110).aspx> on May 13, 2014, 2 pages	
	50	System.Reflection.Emit Namespace, retrieved from <URL: http://msdn.microsoft.com/en-us/library/xd5fw18y(d=default,l=en-us,v=vs.110).aspx> on May 13, 2014, 2 pages	

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature	/HAL SCHNEE/	Date Considered	11/20/2019

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number		15382743
	Filing Date		2016-12-19
	First Named Inventor	Jasmin Cosic	
	Art Unit		15/382,743 - GAU: 2125
	Examiner Name		
	Attorney Docket Number		

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Jasmin Cosic/	Date (YYYY-MM-DD)	2016-12-20
Name/Print	Jasmin Cosic	Registration Number	

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

PTO/SB/08a (01-10)
Approved for use through 07/31/2012. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it contains a valid OMB control number.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS Remove

Examiner Initial*	Cite No	Patent Number	Kind Code1	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	7113946		2006-09-26	Cosic	
	2	7117225		2006-10-03	Cosic	
	3	8335805		2012-12-18	Cosic	
	4	8417740		2013-04-09	Cosic	
	5	8572035		2013-10-29	Cosic	
	6	8655900		2014-02-18	Cosic	
	7	9047324		2015-06-02	Cosic	
	8	9282309		2016-03-08	Cosic	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

	9	9298749		2016-03-29	Cosic	
	10	9367806		2016-06-14	Cosic	
	11	9443192		2016-09-13	Cosic	

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS Remove

Examiner Initial*	Cite No	Publication Number	Kind Code[1]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	20030065662		2003-04-03	Cosic	
	2	20040194017		2004-09-30	Cosic	
	3	20050149517		2005-07-07	Cosic	
	4	20050149542		2005-07-07	Cosic	
	5	20050289105		2005-12-29	Cosic	
	6	20100023541		2010-01-28	Cosic	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

	7	20100082536		2010-04-01	Cosic	
	8	20130218932		2013-08-22	Cosic	
	9	20130226974		2013-08-29	Cosic	
	10	20160140999		2016-05-19	Cosic	
	11	20160142650		2016-05-19	Cosic	
	12	20160246819		2016-08-25	Cosic	
	13	20160246850		2016-08-25	Cosic	
	14	20160246868		2016-08-25	Cosic	
	15	20160292185		2016-10-06	Cosic	

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number[3]	Country Code[2] i	Kind Code[4]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T[5]

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

	1						

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T5
	1		

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature	/HAL SCHNEE/	Date Considered	11/20/2019

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Jasmin Cosic/	Date (YYYY-MM-DD)	2016-12-20
Name/Print	Jasmin Cosic	Registration Number	

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS Remove

Examiner Initial*	Cite No	Patent Number	Kind Code1	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	8996432		2015-03-31	Fu; Jicheng	
	2	6754631		2004-06-22	Din	
	3	9305216		2016-04-05	Mishra	
	4	5560011		1996-09-23	Uyama	
	5	6842877		2005-01-11	Robarts , et al.	
	6	7565340		2009-07-21	Herlocker , et al.	
	7	8261199		2012-09-04	Cradick , et al.	
	8	8266608		2012-09-11	Hecht , et al.	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

	9	9268454		2016-02-23	Hamilton, II , et al.	

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS Remove

Examiner Initial*	Cite No	Publication Number	Kind Code1	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	20120290347		2012-11-15	ELAZOUNI; ASHRAF ; et al.	
	2	20150324685		2015-11-12	Bohn; Richard Esten ; et al.	
	3	20100114746		2010-05-06	Bobbitt; Russell Patrick ; et al.	
	4	20120284026		2012-11-08	Cardillo; Peter S. ; et al.	
	5	20140211988		2014-07-31	Fan; Quanfu ; et al.	
	6	20150006171		2015-01-01	Westby; Michael C. ; et al.	
	7	20150264306		2015-09-17	Marilly; Emmanuel ; et al.	
	8	20110085734		2011-04-14	Berg; Jared S. ; et al.	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

	9	20150269415		2015-09-24	Gelbman; Alexander	
	10	20160274187		2016-09-22	MENON; SANKARAN M. ; et al.	
	11	20150339213		2015-11-22	LEE; Christopher Stephen ; et al.	
	12	20150310041		2015-10-29	Kier; Scott ; et al.	
	13	20160328480		2016-11-10	Owens; Erich James ; et al.	
	14	20140075249		2014-03-13	SATO; Shuhei ; et al.	
	15	20110270794		2011-11-03	Drory; Tal ; et al.	
	16	20070050719		2007-03-01	Lui; Philip ; et al.	
	17	20070061735		2007-03-15	Hoffberg; Steven M. ; et al.	
	18	20060184410		2017-08-17	Ramamurthy; Shankar ; et al.	
	19	20060265406		2006-11-23	Chkodrov; Gueorgui B. ; et al.	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

	20	20090287643		2009-11-19	Corville; Allen O. ; et al.	
	21	20110007079		2011-01-13	Perez et al.	
	22	20070050606		2007-03-01	Ferren et al.	
	23	20090067727		2009-03-12	HIROHATA; Hitoshi	
	24	20130007532		2013-01-03	Miller et al.	
	25	20050240412		2005-10-27	Fujita	
	26	20160167226		2016-06-16	Schnittman	
	27	20090136095		2009-05-28	Marcon et al.	
	28	20080215508		2008-09-04	Hanneman; Jeffrey E. ; et al.	
	29	20090044113		2009-02-12	Jones; Scott T. ; et al.	

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

Examiner Initial*	Cite No	Foreign Document Number3	Country Code2 i	Kind Code4	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T5
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T5
	1	Chen et al. "Case-Based Reasoning System and Artificial Neural Networks: A Review Neural Comput & Applic (2001) 10: pp 264-276, 13 pages	
	2	JOHN J. GREFENSTETTE, CONNIE LOGGIA RAMSEY, ALAN C. SCHULTZ, Learning Sequential Decision Rules Using Simulation Models and Competition, 1990, Navy Center for Applied Research in Artificial Intelligence, Naval Research Laboratory, Washington, DC, 27 pages	
	3	ALAN C. SCHULTZ, JOHN J. GREFENSTETTE, Using a Genetic Algorithm to Learn Behaviors for Autonomous Vehicles, 1992, Navy Center for Applied Research in Artificial Intelligence, Naval Research Laboratory, Washington, DC, 12 pages	
	4	Koppula et al., "Anticipating human activities using object affordances for reactive robotic response", IEEE TRAMI 2016, published 5 May 2015, 16 pages	
	5	Orme, "System design tips for entry level smartphones - part 3", found online at "https://community.arm.com/processors/b/blog/posts/system-design-tips-for-entry-level-smartphones---part-3", Oct 21, 2013, 12 pages	
	6	NOWOSTAWSKI, MARIUSZ et al., "Dynamic Demes Parallel Genetic Algorithm", May 13, 1999, 6 pages	
	7	Stack Overflow, How can I quantify difference between two images?, accessed 2 August 2017 at https://stackoverflow.com/questions/189943/how-can-i-quantify-difference-between-two-images, 8 pages	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

	8	SKOLICKI, ZBIGNIEW et al., "The Influence of Migration Sizes and Intervals on Island Models", June 29, 2005, 8 pages	
	9	Vahid Lari, et al., "Decentralized dynamic resource management support for massively parallel processor arrays", September 11, 2011, 8 pages	
	10	Vahid Lari, et al., "Distributed resource reservation in massively parallel processor arrays", May 16, 2011, 4 pages	
	11	Mohsen Hayati and Yazdan Shirvany, "Artificial Neural Network Approach for Short Term Load Forecasting for Illam Region", January 2007, 5 pages	
	12	Hasim Sak, Andrew Senior, and Francoise Beaufays, "Long Short-Term Memory Recurrent Neural Network Architectures for Large Scale Acoustic Modeling", January 2014, 5 pages	
	13	Jorg Walter and Klaus Schulten, "Implementation of self-organizing neural networks for visuo-motor control of an industrial robot", January 1993, 10 pages	
	14	Heikki Hyotyniemi and Aarno Lehtola, "A Universal Relation Database Interface for Knowledge Based Systems", April 1991, 5 pages	
	15	Mrissa, Michael, et al. "An avatar architecture for the web of things." IEEE Internet Computing 19.2 (2015): 30-38., 9 pages	
	16	Luck, Michael, and Ruth Aylett. "Applying artificial intelligence to virtual reality: Intelligent virtual environments." Applied Artificial Intelligence 14.1 (2000): 3-32., 30 pages	
	17	Terdjimi, Mehdi, et al. "An avatar-based adaptation workflow for the web of things." Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), 2016 IEEE 25th International Conference on. IEEE, 2016., 6 pages	
	18	Bogdanovych, Anton, et al. "Authentic interactive reenactment of cultural heritage with 3D virtual worlds and artificial intelligence." Applied Artificial Intelligence 24.6 (2010): 617-647., 32 pages	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

	19	Hernandez, Marco E. Perez, and Stephan Reiff-Marganiec. "Autonomous and self controlling smart objects for the future internet." Future internet of things and cloud (FiCloud), 2015 3rd international conference on. IEEE, 2015., 8 pages	
	20	Medini et al., "Building a Web of Things with Avatars", Managing the Web of Things (2017), 30 pages	

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature	/HAL SCHNEE/	Date Considered	11/20/2019

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Jasmin Cosic/	Date (YYYY-MM-DD)	2019-07-02
Name/Print	Jasmin Cosic	Registration Number	

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

PTO/SB/08a (01-10)
Approved for use through 07/31/2012. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it contains a valid OMB control number.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS

Remove

Examiner Initial*	Cite No	Patent Number	Kind Code1	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1					

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS

Remove

Examiner Initial*	Cite No	Publication Number	Kind Code1	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1					

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS

Remove

Examiner Initial*	Cite No	Foreign Document Number3	Country Code2 i	Kind Code4	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T5
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS

Remove

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T5

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

	1	Tracing (software), retrieved from <URL: http://wikipedia.com> on Jan 10, 2014, 3 pages	
	2	Tree (data structure), retrieved from <URL: http://wikipedia.com> on Jun 24, 2014, 6 pages	
	3	PTRACE(2), retrieved from <URL: http://unixhelp.ed.ac.uk/CGI/man-cgi?ptrace> on Mar 19, 2014, 5 pages	
	4	Wevtutil, retrieved from <URL: http://technet.microsoft.com/en-us/library/cc732848(d=default,l=en-us,v=ws.11).aspx> on Apr 28, 2014, 5 pages	
	5	Intel Processor Trace, retrieved from <URL: https://software.intel.com/en-us/blogs/2013/09/18/processor-tracing> on Apr 28, 2014, 3 pages	
	6	YOUNGHOON JUNG, JAVA DYNAMICS Reflection and a lot more, Oct 10, 2012, 55 pages, Columbia University	
	7	AMITABH SRIVASTAVA, ALAN EUSTACE, ATOM A System for Building Customized Program Analysis Tools, May 3, 2004, 12 pages	
	8	MATHEW SMITHSON, KAPIL ANAND, APARNA KOTHA, KHALED ELWAZEER, NATHAN GILES, RAJEEV BARUA, Binary Rewriting without Relocation Information, Nov 10, 2010, 11 pages, University of Maryland	
	9	MAREK OLSZEWSKI, KEIR MIERTE, ADAM CZAJKOWSKI, ANGELA DEMLE BROWN, JIT Instrumentation - A Novel Approach To Dynamically Instrument Operating Systems, Feb 12, 2007, 14 pages, University of Toronto	

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature	/HAL SCHNEE/	Date Considered	11/20/2019

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number		15382743
	Filing Date		2016-12-19
	First Named Inventor	Jasmin Cosic	
	Art Unit		15/382,743 - GAU: 2125
	Examiner Name		
	Attorney Docket Number		

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Jasmin Cosic/	Date (YYYY-MM-DD)	2016-12-20
Name/Print	Jasmin Cosic	Registration Number	

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

PTO/SB/08a (01-10)
Approved for use through 07/31/2012. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

U.S.PATENTS Remove

Examiner Initial*	Cite No	Patent Number	Kind Code1	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	5758333		1998-05-26	Bauer , et al.	
	2	7409401		2008-08-05	Hansen , et al.	
	3	7533128		2009-05-12	Sanchez , et al.	
	4	7831564		2010-11-09	Wei , et al.	
	5	7444338		2008-10-28	Fisher	
	6	7082435		2006-07-25	Guzman , et al.	
	7	8949186		2015-02-03	Yueh , et al.	
	8	8762428		2014-06-24	Kulack , et al.	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

	9	7395255		2008-07-01	Li	
	10	6850942		2005-02-01	Cotner , et al.	
	11	5983232		1999-11-09	Zhang	
	12	5592661		1997-01-07	Eisenberg , et al.	
	13	7849114		2010-12-07	Boss , et al.	

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS Remove

Examiner Initial*	Cite No	Publication Number	Kind Code[1]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	20120089570		2012-04-12	Zha; Charles Li ; et al.	
	2	20140143276		2014-05-22	ROGERS; Michael Anthony ; et al.	
	3	20080071770		2008-03-20	Schloter; Philipp ; et al.	
	4	20110004586		2011-01-06	Cherryholmes; Lon Jones ; et al.	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

	5	20050154745		2005-07-14	Hansen, Lynda A. ; et al.	
	6	20140164430		2014-06-12	Hadjieleftheriou; Marios ; et al.	
	7	20020019822		2002-02-14	Seki, Yumiko ; et al.	
	8	20130204907		2013-08-08	Alonso Alarcon; Antonio ; et al.	
	9	20060259466		2006-11-16	Bilotti; David	
	10	20110093435		2011-04-21	Zha; Charlie Li ; et al.	
	11	20090070337		2009-03-12	Romem; Yaniv ; et al.	

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number3	Country Code2 i	Kind Code4	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T5
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS Remove

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT
(Not for submission under 37 CFR 1.99)

Application Number	15382743
Filing Date	2016-12-19
First Named Inventor	Jasmin Cosic
Art Unit	15/382,743 - GAU: 2125
Examiner Name	
Attorney Docket Number	

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T5
	1	2D computer graphics, retrieved from <URL: http://wikipedia.com> on Nov 9, 2014, 7 pages	
	2	3D computer graphics, retrieved from <URL: http://wikipedia.com> on Nov 9, 2014, 3 pages	
	3	3D modeling, retrieved from <URL: http://wikipedia.com> on Nov 9, 2014, 5 pages	
	4	8: Mouse Handling, retrieved from <URL: http://ericsink.com/wpf3d/8_Mouse.html> on Nov 13, 2014, 2 pages	
	5	AutoCAD, retrieved from <URL: http://wikipedia.com> on Nov 9, 2014, 7 pages	
	6	Collision detection, retrieved from <URL: http://wikipedia.com> on Nov 23, 2014, 5 pages	
	7	Comparison of OpenGL and Direct3D, retrieved from <URL: http://wikipedia.com> on Nov 22, 2014, 7 pages	
	8	Data dictionary, retrieved from <URL: http://wikipedia.com> on Nov 6, 2014, 3 pages	
	9	DBA_TABLES, retrieved from <URL: http://wikipedia.com> on Nov 6, 2014, 1 pages	
	10	Digital sculpting, retrieved from <URL: http://wikipedia.com> on Nov 12, 2014, 3 pages	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

	11	Facial recognition system, retrieved from <URL: http://wikipedia.com> on Dec 13, 2014, 7 pages	
	12	Game engine, retrieved from <URL: http://wikipedia.com> on Nov 23, 2014, 4 pages	
	13	JavaFX, retrieved from <URL: http://wikipedia.com> on Nov 22, 2014, 7 pages	
	14	Jreality, retrieved from <URL: http://wikipedia.com> on Nov 22, 2014, 3 pages	
	15	List of 14+ Image Recognition APIs, retrieved from <URL: http://blog.mashape.com/list-of-14-image-recognition-apis/> on Dec 13, 2014, 2 pages	
	16	List of 50+ Face Detection / Recognition APIs, libraries, and software, retrieved from <URL: http://blog.mashape.com/list-of-50-face-detection-recognition-apis/> on Dec 13, 2014, 4 pages	
	17	List of file formats, retrieved from <URL: http://wikipedia.com> on Nov 9, 2014, 33 pages	
	18	List of game engines, retrieved from <URL: http://wikipedia.com> on Nov 22, 2014, 17 pages	
	19	Lua (programming language), retrieved from <URL: http://wikipedia.com> on Nov 25, 2014, 10 pages	
	20	Mantle (API), retrieved from <URL: http://wikipedia.com> on Nov 22, 2014, 5 pages	
	21	Microsoft XNA, retrieved from <URL: http://wikipedia.com> on Nov 22, 2014, 7 pages	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT
(Not for submission under 37 CFR 1.99)

Application Number	15382743
Filing Date	2016-12-19
First Named Inventor	Jasmin Cosic
Art Unit	15/382,743 - GAU: 2125
Examiner Name	
Attorney Docket Number	

	22	3.4 Getting Information About Databases and Tables, retrieved from <URL: http://dev.mysql.com/doc/refman/5.7/en/getting-information.html> on Nov 6, 2014, 2 pages	
	23	MySQL show users - how to show the users in a MySQL database, retrieved from <URL: http://alvinalexander.com/blog/post/mysql/show-users-i-ve-created-in-mysql-database> on Nov 6, 2014, 4 pages	
	24	OpenCV, retrieved from <URL: http://wikipedia.com> on Dec 13, 2014, 3 pages	
	25	Optical character recognition, retrieved from <URL: http://wikipedia.com> on Dec 13, 2014, 6 pages	
	26	Outline of object recognition, retrieved from <URL: http://wikipedia.com> on Dec 13, 2014, 8 pages	
	27	Particle system, retrieved from <URL: http://wikipedia.com> on Nov 12, 2014, 3 pages	
	28	Physics engine, retrieved from <URL: http://wikipedia.com> on Nov 23, 2014, 5 pages	
	29	Polygonal modeling, retrieved from <URL: http://wikipedia.com> on Nov 12, 2014, 4 pages	
	30	POV-Ray, retrieved from <URL: http://wikipedia.com> on Nov 10, 2014, 5 pages	
	31	Raster graphics, retrieved from <URL: http://wikipedia.com> on Nov 9, 2014, 3 pages	
	32	Scene graph, retrieved from <URL: http://wikipedia.com> on Nov 22, 2014, 4 pages	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

	33	STL (file format), retrieved from <URL: http://wikipedia.com> on Nov 11, 2014, 4 pages	
	34	Subdivision surface, retrieved from <URL: http://wikipedia.com> on Nov 12, 2014, 3 pages	
	35	sysobjects, retrieved from <URL: http://technet.microsoft.com/en-us/library/aa260447(d=default,l=en-us,v=sql.80).aspx> on Nov 6, 2014, 2 pages	
	36	Vector graphics, retrieved from <URL: http://wikipedia.com> on Nov 9, 2014, 5 pages	
	37	Vertex (computer graphics), retrieved from <URL: http://wikipedia.com> on Nov 9, 2014, 2 pages	
	38	Voxel, retrieved from <URL: http://wikipedia.com> on Nov 11, 2014, 5 pages	
	39	Wavefront .obj file, retrieved from <URL: http://wikipedia.com> on Nov 9, 2014, 6 pages	
	40	Acrobat 3D tutorials - basic interaction, retrieved from <URL: https://acrobatusers.com/assets/collections/tutorials/legacy/tech_corners/3d/3d_tutorials/basic_interaction.pdf> on Nov 23, 2014, 2 pages	

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature	/HAL SCHNEE/	Date Considered	11/20/2019

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Jasmin Cosic/	Date (YYYY-MM-DD)	2016-12-20
Name/Print	Jasmin Cosic	Registration Number	

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

PART B - FEE(S) TRANSMITTAL

Complete and send this form, together with applicable fee(s), by mail or fax, or via EFS-Web.

By mail, send to: Mail Stop ISSUE FEE
Commissioner for Patents
P.O. Box 1450
Alexandria, Virginia 22313-1450

By fax, send to: (571)-273-2885

INSTRUCTIONS: This form should be used for transmitting the ISSUE FEE and PUBLICATION FEE (if required). Blocks 1 through 5 should be completed where appropriate. All further correspondence including the Patent, advance orders and notification of maintenance fees will be mailed to the current correspondence address as indicated unless corrected below or directed otherwise in Block 1, by (a) specifying a new correspondence address; and/or (b) indicating a separate "FEE ADDRESS" for maintenance fee notifications.

CURRENT CORRESPONDENCE ADDRESS (Note: Use Block 1 for any change of address)

116094 7590 12/11/2019

Jasmin Cosic
108 Woodbury Street
Pawtucket, RI 02861

Note: A certificate of mailing can only be used for domestic mailings of the Fee(s) Transmittal. This certificate cannot be used for any other accompanying papers. Each additional paper, such as an assignment or formal drawing, must have its own certificate of mailing or transmission.

Certificate of Mailing or Transmission

I hereby certify that this Fee(s) Transmittal is being deposited with the United States Postal Service with sufficient postage for first class mail in an envelope addressed to the Mail Stop ISSUE FEE address above, or being transmitted to the USPTO via EFS-Web or by facsimile to (571) 273-2885, on the date below.

Jasmin Cosic	(Typed or printed name)
/Jasmin Cosic/	(Signature)
February 20, 2020	(Date)

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
15/382,743	12/19/2016	Jasmin Cosic		4742

TITLE OF INVENTION: ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING AN AVATAR'S CIRCUMSTANCES FOR AUTONOMOUS AVATAR OPERATION

APPLN. TYPE	ENTITY STATUS	ISSUE FEE DUE	PUBLICATION FEE DUE	PREV. PAID ISSUE FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	SMALL	$500	$0.00	$0.00	$500	03/11/2020

EXAMINER	ART UNIT	CLASS-SUBCLASS
SCHNEE, HAL W	2125	706-011000

1. Change of correspondence address or indication of "Fee Address" (37 CFR 1.363).

☐ Change of correspondence address (or Change of Correspondence Address form PTO/SB/122) attached.

☐ "Fee Address" indication (or "Fee Address" Indication form PTO/SB/47; Rev 03-09 or more recent) attached. **Use of a Customer Number is required.**

2. For printing on the patent front page, list
(1) The names of up to 3 registered patent attorneys or agents OR, alternatively,
(2) The name of a single firm (having as a member a registered attorney or agent) and the names of up to 2 registered patent attorneys or agents. If no name is listed, no name will be printed.

1 ____________________
2 ____________________
3 ____________________

3. ASSIGNEE NAME AND RESIDENCE DATA TO BE PRINTED ON THE PATENT (print or type)

PLEASE NOTE: Unless an assignee is identified below, no assignee data will appear on the patent. If an assignee is identified below, the document must have been previously recorded, or filed for recordation, as set forth in 37 CFR 3.11 and 37 CFR 3.81(a). Completion of this form is NOT a substitute for filing an assignment.

(A) NAME OF ASSIGNEE

(B) RESIDENCE: (CITY and STATE OR COUNTRY)

Please check the appropriate assignee category or categories (will not be printed on the patent) : ☐ Individual ☐ Corporation or other private group entity ☐ Government

4a. Fees submitted: ☒ Issue Fee ☐ Publication Fee (if required) ☐ Advance Order - # of Copies ____________

4b. Method of Payment: *(Please first reapply any previously paid fee shown above)*

☒ Electronic Payment via EFS-Web ☐ Enclosed check ☐ Non-electronic payment by credit card (Attach form PTO-2038)

☐ The Director is hereby authorized to charge the required fee(s), any deficiency, or credit any overpayment to Deposit Account No. __________

5. **Change in Entity Status** (from status indicated above)

☐ Applicant certifying micro entity status. See 37 CFR 1.29

☐ Applicant asserting small entity status. See 37 CFR 1.27

☐ Applicant changing to regular undiscounted fee status.

NOTE: Absent a valid certification of Micro Entity Status (see forms PTO/SB/15A and 15B), issue fee payment in the micro entity amount will not be accepted at the risk of application abandonment.
NOTE: If the application was previously under micro entity status, checking this box will be taken to be a notification of loss of entitlement to micro entity status.
NOTE: Checking this box will be taken to be a notification of loss of entitlement to small or micro entity status, as applicable.

NOTE: This form must be signed in accordance with 37 CFR 1.31 and 1.33. See 37 CFR 1.4 for signature requirements and certifications.

Authorized Signature /Jasmin Cosic/ Date February 20, 2020

Typed or printed name Jasmin Cosic Registration No. ____________

Electronic Patent Application Fee Transmittal

Application Number:	15382743
Filing Date:	19-Dec-2016
Title of Invention:	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING AN AVATAR'S CIRCUMSTANCES FOR AUTONOMOUS AVATAR OPERATION
First Named Inventor/Applicant Name:	Jasmin Cosic
Filer:	Jasmin Cosic
Attorney Docket Number:	

Filed as Small Entity

Filing Fees for **Utility under 35 USC 111(a)**

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Basic Filing:				
Pages:				
Claims:				
Miscellaneous-Filing:				
Petition:				
Patent-Appeals-and-Interference:				
Post-Allowance-and-Post-Issuance:				
UTILITY APPL ISSUE FEE	2501	1	500	500

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Extension-of-Time:				
Miscellaneous:				
	Total in USD ($)			**500**

Electronic Acknowledgement Receipt

EFS ID:	38648739
Application Number:	15382743
International Application Number:	
Confirmation Number:	4742
Title of Invention:	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING AN AVATAR'S CIRCUMSTANCES FOR AUTONOMOUS AVATAR OPERATION
First Named Inventor/Applicant Name:	Jasmin Cosic
Customer Number:	116094
Filer:	Jasmin Cosic
Filer Authorized By:	
Attorney Docket Number:	
Receipt Date:	20-FEB-2020
Filing Date:	19-DEC-2016
Time Stamp:	18:32:07
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	yes
Payment Type	CARD
Payment was successfully received in RAM	$500
RAM confirmation Number	E20202JI34362290
Deposit Account	
Authorized User	

The Director of the USPTO is hereby authorized to charge indicated fees and credit any overpayment as follows:

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1	Issue Fee Payment (PTO-85B)	PTOL-85B_Issue_Fee_Form.pdf	2774029 541ffd1000f2c163b82d99de01966e2543158179	no	1
Warnings:					
Information:					
2	Fee Worksheet (SB06)	fee-info.pdf	30013 00bf8f073689e5b26b76c8fc3f7cdbb1ee01665a	no	2
Warnings:					
Information:					
		Total Files Size (in bytes):	2804042		

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.

National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15382743
	Filing Date	2016-12-19
	First Named Inventor	Jasmin Cosic
	Art Unit	15/382,743 - GAU: 2125
	Examiner Name	
	Attorney Docket Number	

	9	20150269415		2015-09-24	Gelbman; Alexander	
	10	20160274187		2016-09-22	MENON; SANKARAN M. ; et al.	
	11	20150339213		2015-11-22	LEE; Christopher Stephen ; et al.	
	12	20150310041		2015-10-29	Kier; Scott ; et al.	
	13	20160328480		2016-11-10	Owens; Erich James ; et al.	
	14	20140075249		2014-03-13	SATO; Shuhei ; et al.	
	15	20110270794		2011-11-03	Drory; Tal ; et al.	
	16	20070050719		2007-03-01	Lui; Philip ; et al.	
	17	20070061735		2007-03-15	Hoffberg; Steven M. ; et al.	
Change(s) applied to document,	18	20060184410		2006 ~~2017~~-08-17	Ramamurthy; Shankar ; et al.	
/C.L./ 2/3/2020	19	20060265406		2006-11-23	Chkodrov; Gueorgui B. ; et al.	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /H.S./

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	ISSUE DATE	PATENT NO.	ATTORNEY DOCKET NO.	CONFIRMATION NO.
15/382,743	03/31/2020	10607134		4742

116094 7590 03/11/2020

Jasmin Cosic
108 Woodbury Street
Pawtucket, RI 02861

ISSUE NOTIFICATION

The projected patent number and issue date are specified above.

Determination of Patent Term Adjustment under 35 U.S.C. 154 (b)
(application filed on or after May 29, 2000)

The Patent Term Adjustment is 763 day(s). Any patent to issue from the above-identified application will include an indication of the adjustment on the front page.

If a Continued Prosecution Application (CPA) was filed in the above-identified application, the filing date that determines Patent Term Adjustment is the filing date of the most recent CPA.

Applicant will be able to obtain more detailed information by accessing the Patent Application Information Retrieval (PAIR) WEB site (http://pair.uspto.gov).

Any questions regarding the Patent Term Extension or Adjustment determination should be directed to the Office of Patent Legal Administration at (571)-272-7702. Questions relating to issue and publication fee payments should be directed to the Application Assistance Unit (AAU) of the Office of Data Management (ODM) at (571)-272-4200.

APPLICANT(s) (Please see PAIR WEB site http://pair.uspto.gov for additional applicants):

Jasmin Cosic, Miami, FL;